#WWDC18

Adding Delight to Your iOS App

Six pro tips to make it magic

Session 233

Ben Fearnley, iOS System Experience
Peter Hajas, iOS System Experience

© 2018 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.

External Display Support

Layout-Driven UI

Laser-Fast Launches

Smooth Scrolling

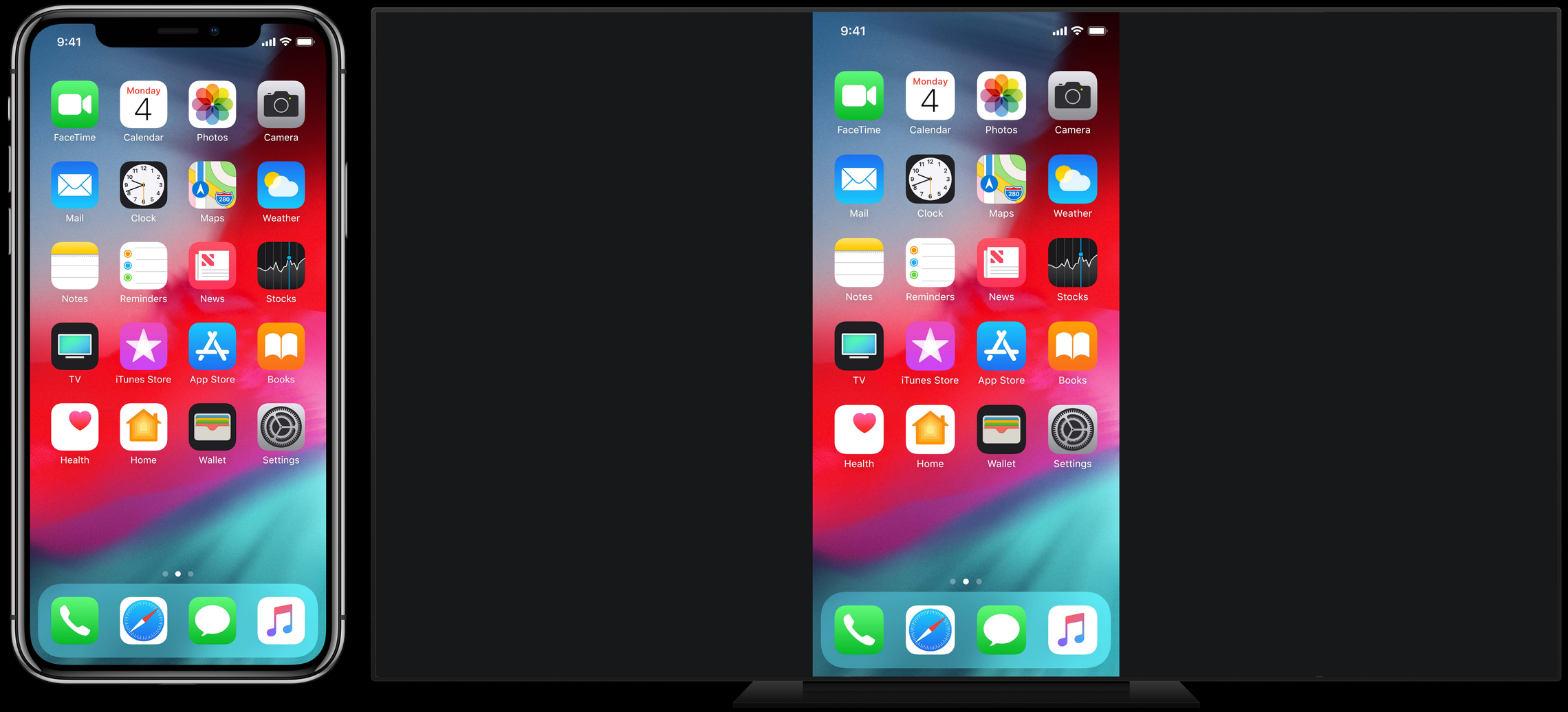
Continuing with Continuity

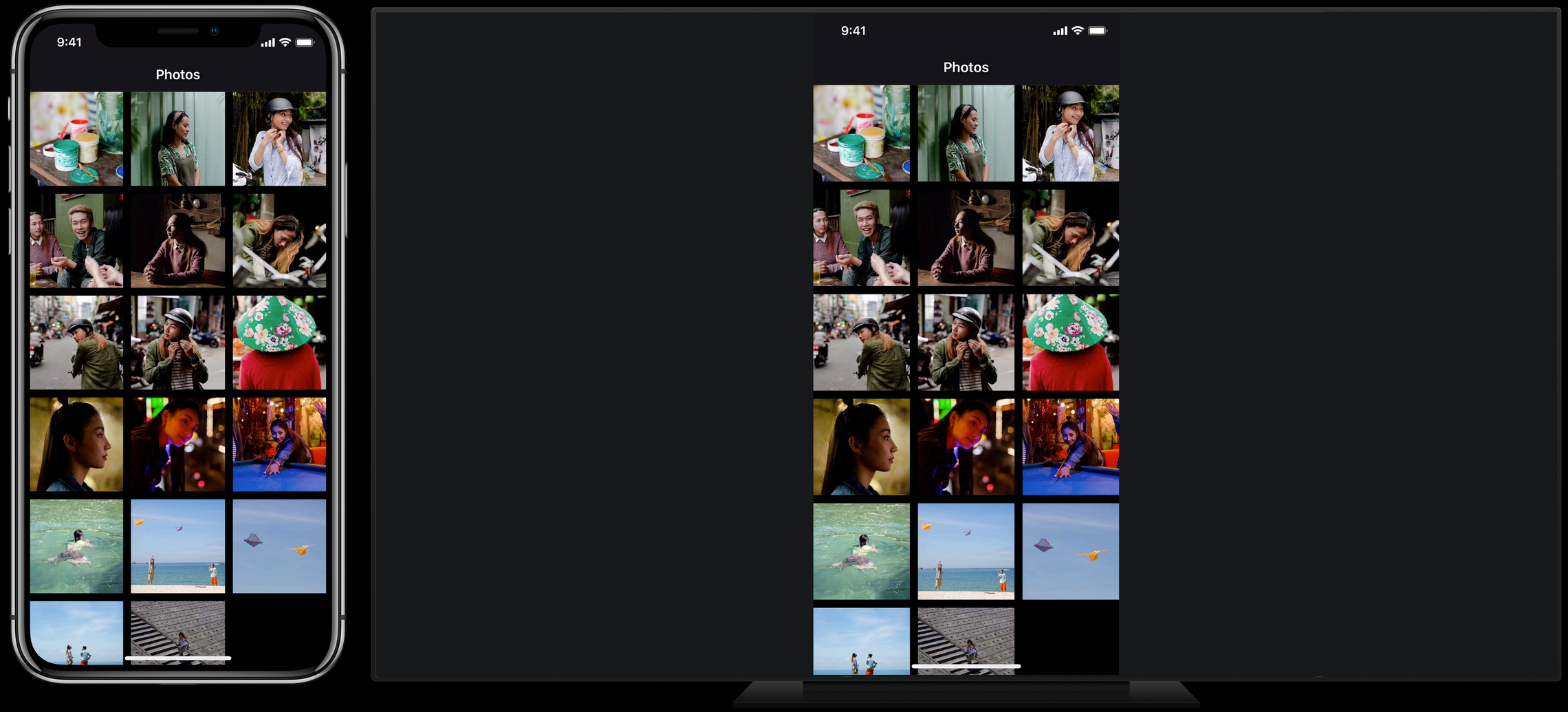
Debugging Like a Pro

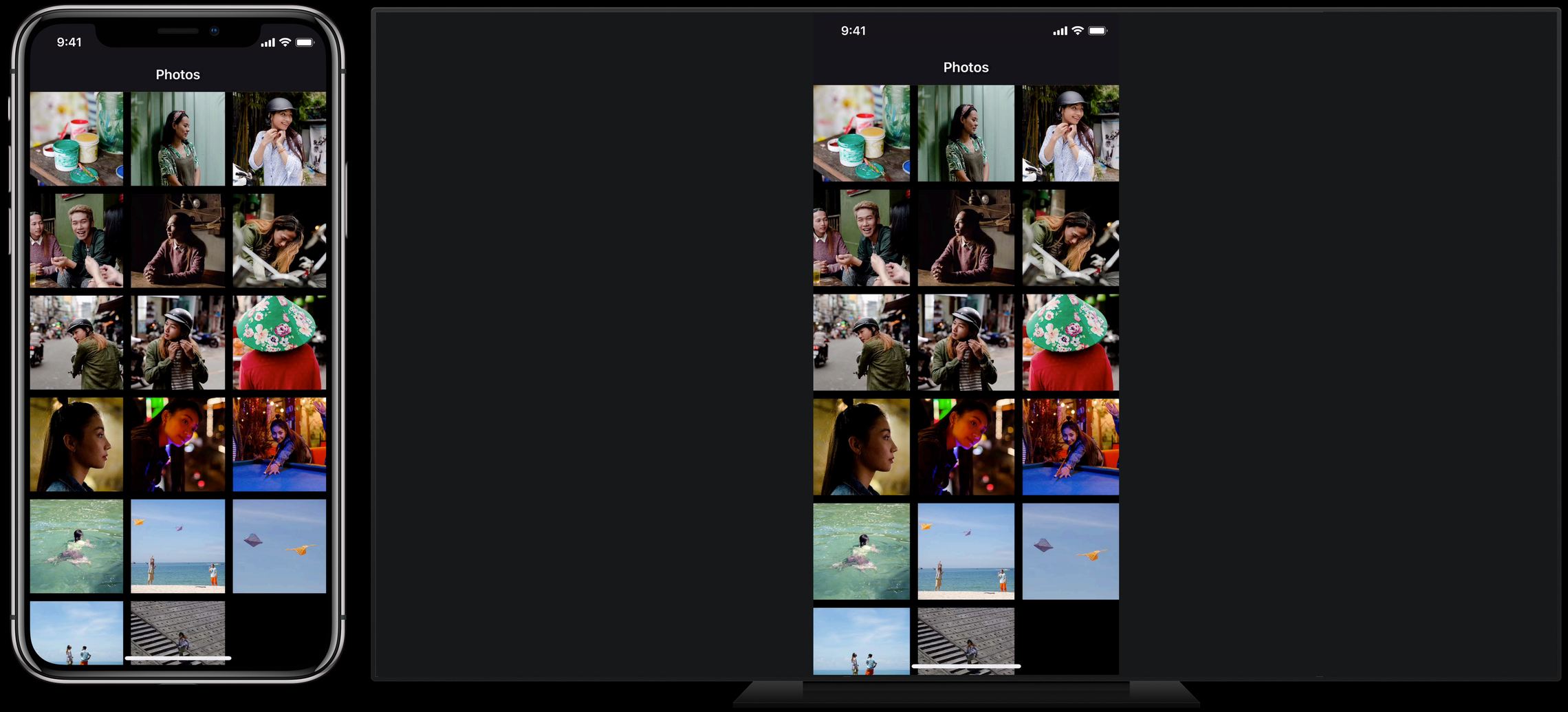
External Display Support

Ben Fearnley, iOS System Experience
Peter Hajas, iOS System Experience

9:41
FaceTime
Monday
4
Calendar
Photos
Camera
Mail
Clock
Maps
280
Weather
Notes
Reminders
News
Stocks
TV
iTunes Store
App Store
Books
Health
Home
Wallet
Settings


9:41
FaceTime
Monday
4
Calendar
Photos
Camera
Mail
Clock
Maps
Weather
Notes
Reminders
News
Stocks
TV
iTunes Store
App Store
Books
Health
Home
Wallet
Settings
9:41
FaceTime
Monday
4
Calendar
Photos
Camera
Mail
Clock
Maps
Weather
Notes
Reminders
News
Stocks
TV
iTunes Store
App Store
Books
Health
Home
Wallet
Settings


9:41
Photos
9:41
Photos


9:41
Photos
9:41
Photos

Photos
A Great Photo
Photos
A Great Photo

We can do better.

#WWDC18

Adding Delight to Your iOS App

Six pro tips to make it magic

Session 233

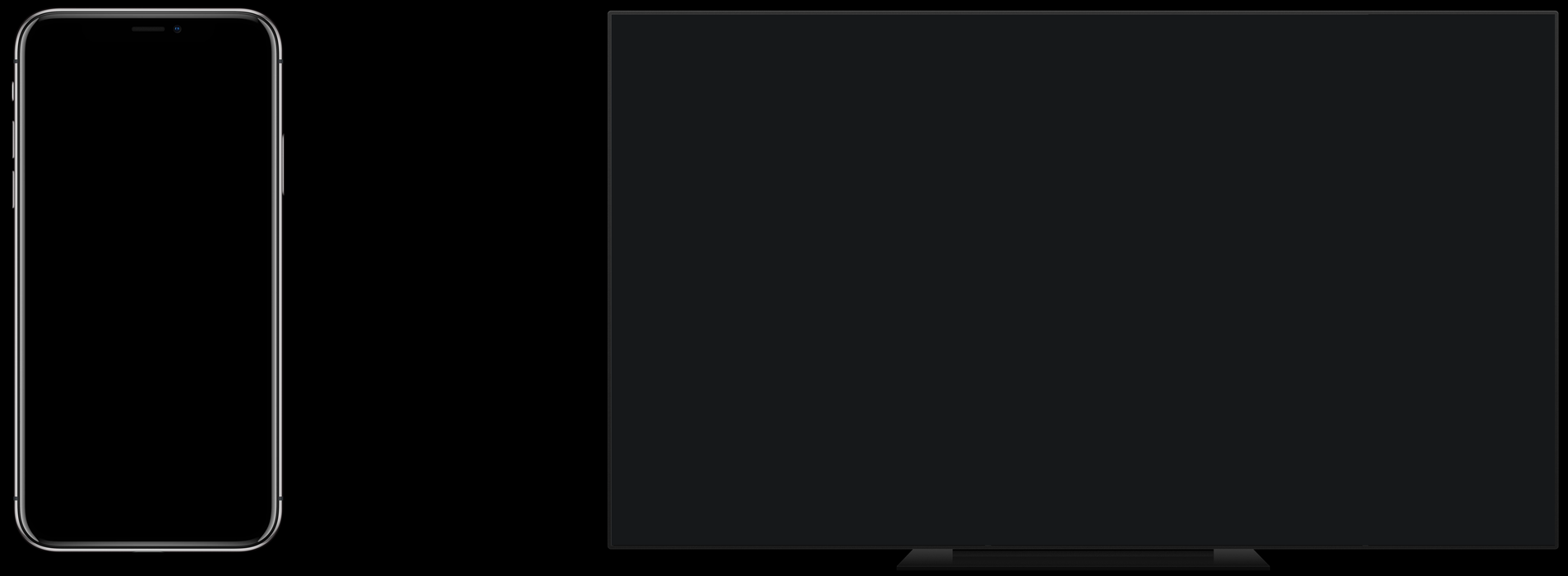
Ben Fearnley, iOS System Experience
Peter Hajas, iOS System Experience

© 2018 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.

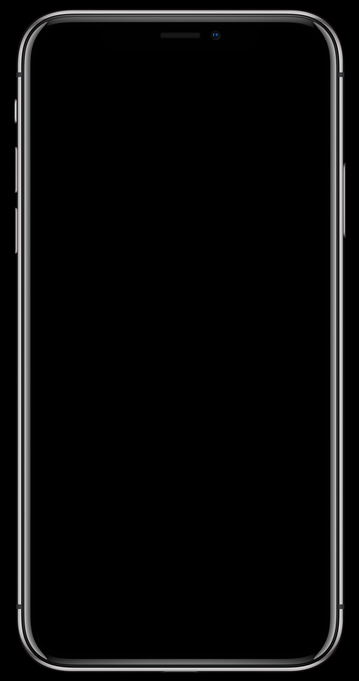
APPLE TV CONTROLLER
Device connected.
Brake
Eric Song
4
60 km/h
Tiesha

Designing for Multiple Displays

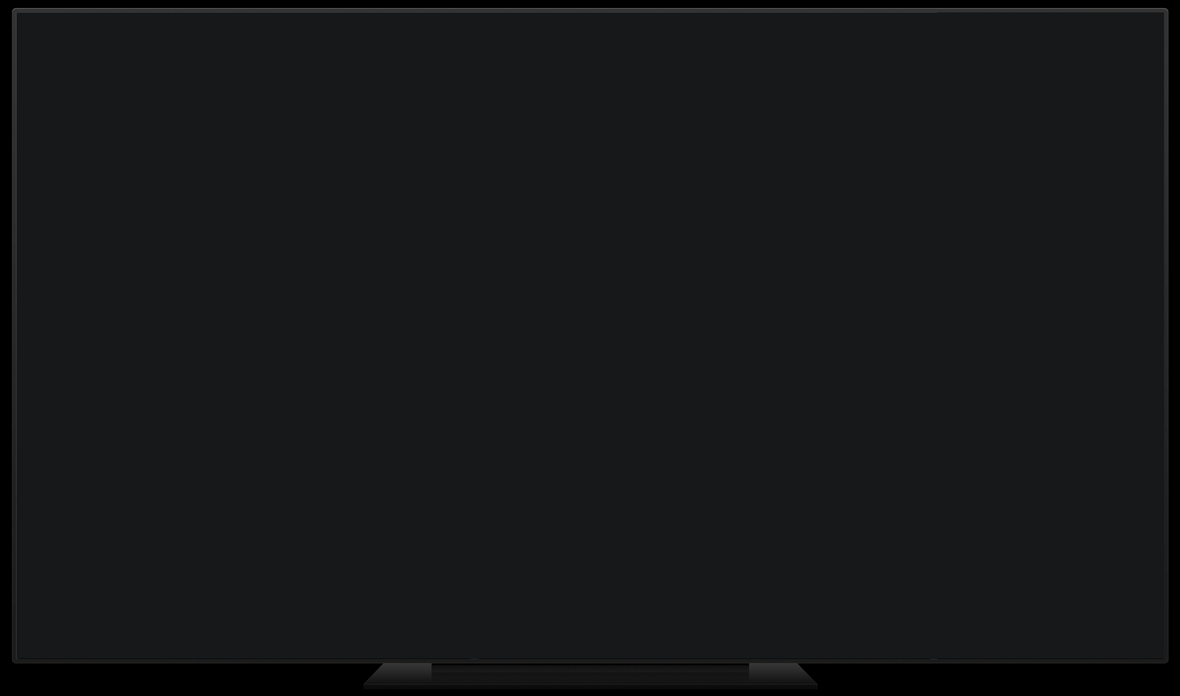
Designing for Multiple Displays



Designing for Multiple Displays

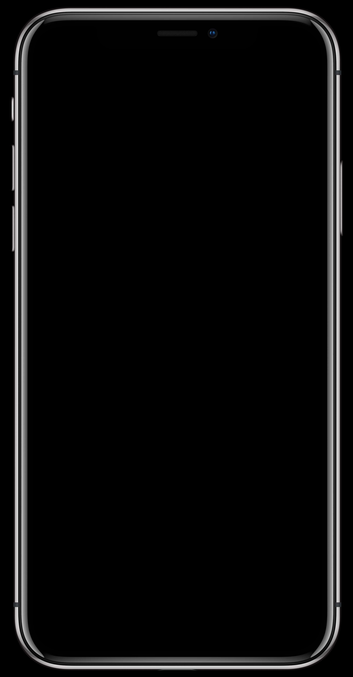


Small



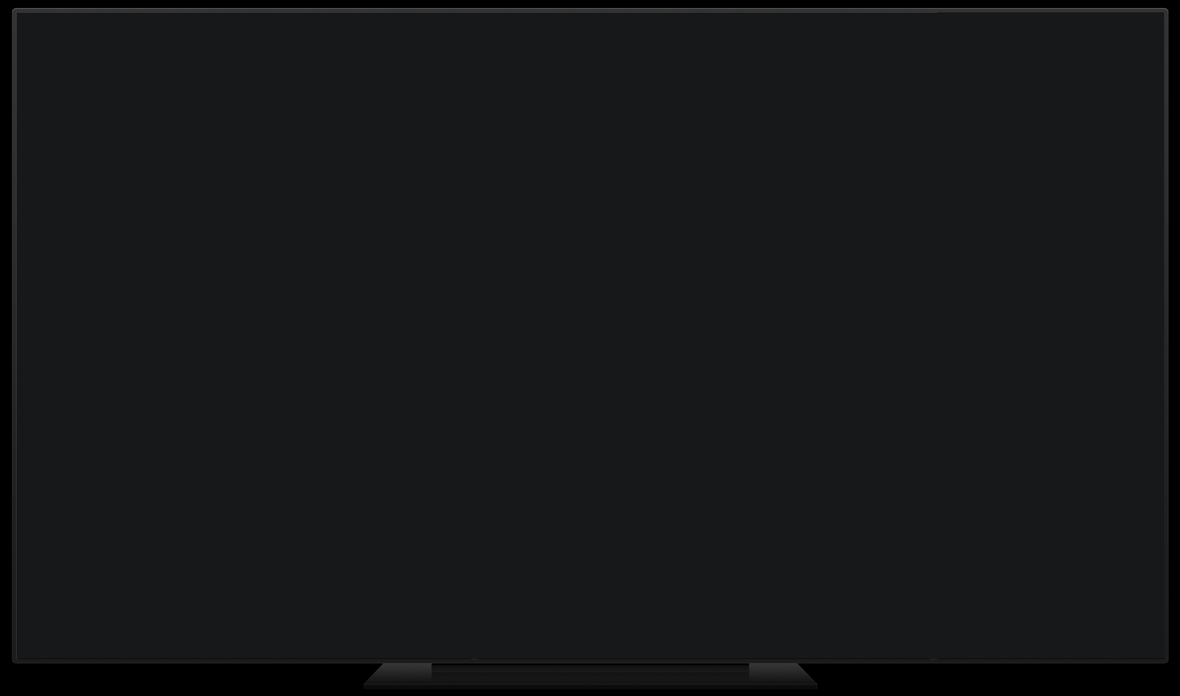
Large

Designing for Multiple Displays



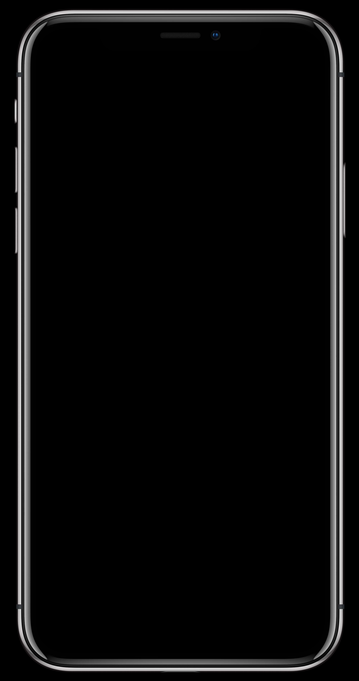
Small

Private



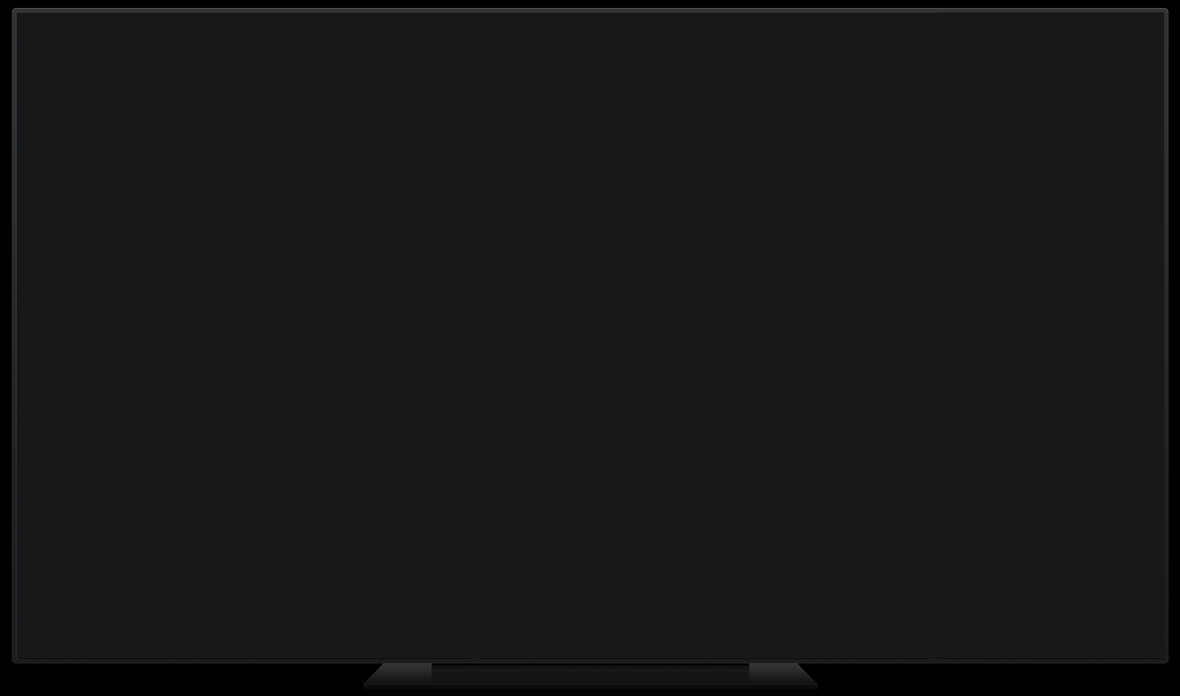
Large

Designing for Multiple Displays



Small

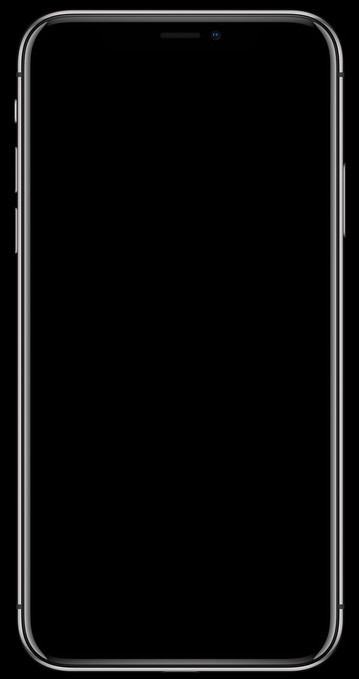
Private



Large

Public

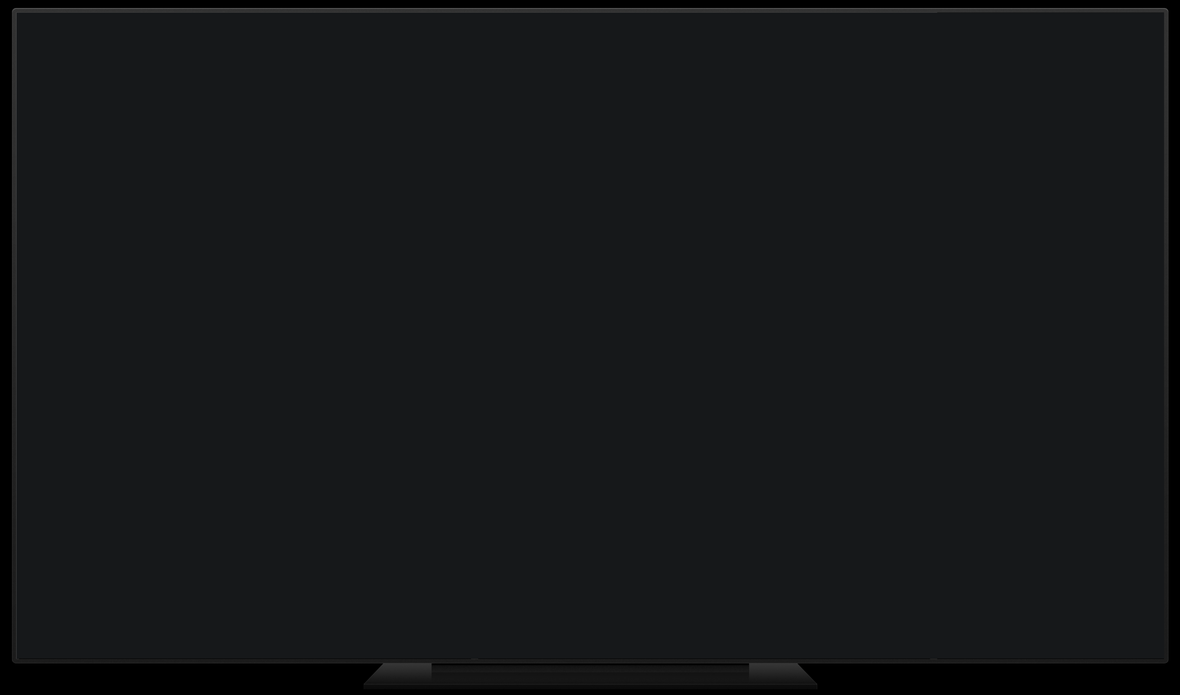
Designing for Multiple Displays



Small

Private

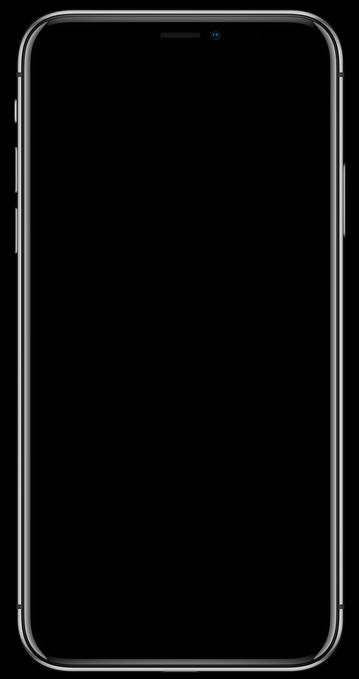
Interactive

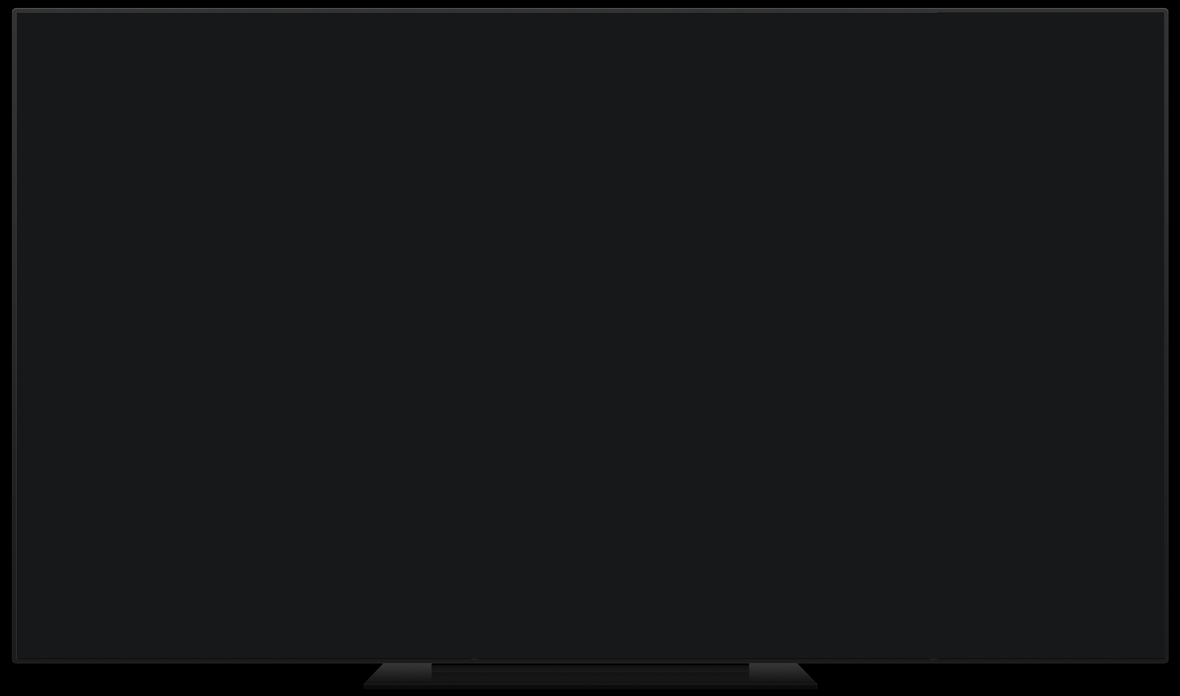


Large

Public

Designing for Multiple Displays





Small

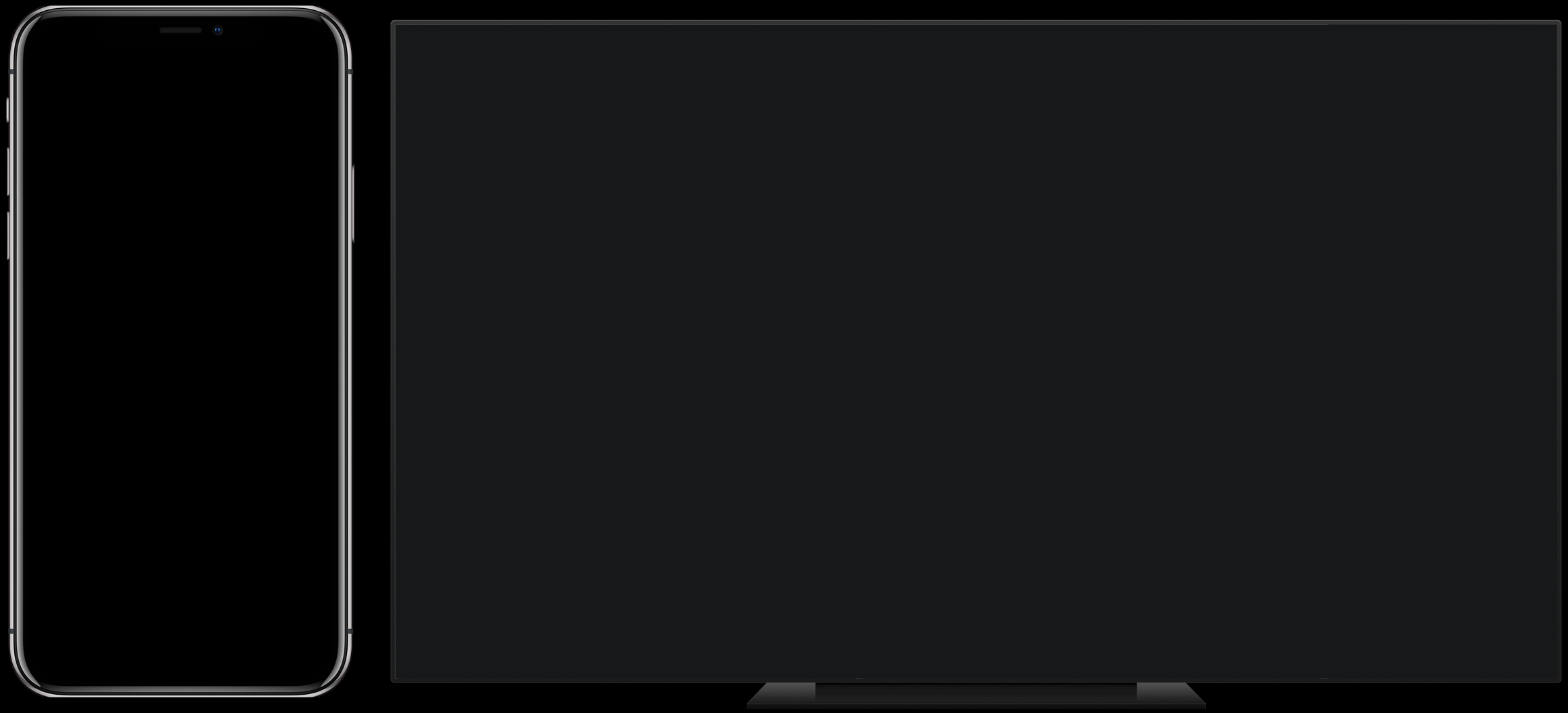
Private

Interactive

Large

Public

Non-Interactive



9:41
Photos

9:41
Photos

9:41
Photos

Connectivity

Behavior

Transitions

Handling Connectivity

Is there an external display connected?

```
class var screens: [UIScreen]
```

```
if UIScreen.screens.count > 1 {
    // External display is connected
    ...
}
```

Handling Connectivity

What happens when the display is connected or disconnected?

```
UIScreen.didConnectNotification
UIScreen.didDisconnectNotification
```

```swift
if let externalScreen = UIScreen.screens.last {
    externalWindow = UIWindow()
    externalWindow.screen = externalScreen
    configureExternalWindow(externalWindow)
    externalWindow.isHidden = false
}
```

```swift
if let externalScreen = UIScreen.screens.last {
    externalWindow = UIWindow()
    externalWindow.screen = externalScreen
    configureExternalWindow(externalWindow)
    externalWindow.isHidden = false
}
```

```
if let externalScreen = UIScreen.screens.last {
    externalWindow = UIWindow()
    externalWindow.screen = externalScreen
    configureExternalWindow(externalWindow)
    externalWindow.isHidden = false
}
```

```
if let externalScreen = UIScreen.screens.last {
    externalWindow = UIWindow()
    externalWindow.screen = externalScreen
    configureExternalWindow(externalWindow)
    externalWindow.isHidden = false
}
```

```swift
if let externalScreen = UIScreen.screens.last {
    externalWindow = UIWindow()
    externalWindow.screen = externalScreen
    configureExternalWindow(externalWindow)
    externalWindow.isHidden = false
}
```

```
externalWindow.isHidden = true
externalWindow = nil
```

```
externalWindow.isHidden = true
externalWindow = nil
```

Change Your App’s Behavior

Change Your App's Behavior

```
// In our Collection View selection callback
if inSingleDisplayMode {
    photoViewController.photo = photo
    navigationController?.pushViewController(photoViewController, animated: true)
} else {
    showOnExternalDisplay(photo)
}
```

Change Your App's Behavior

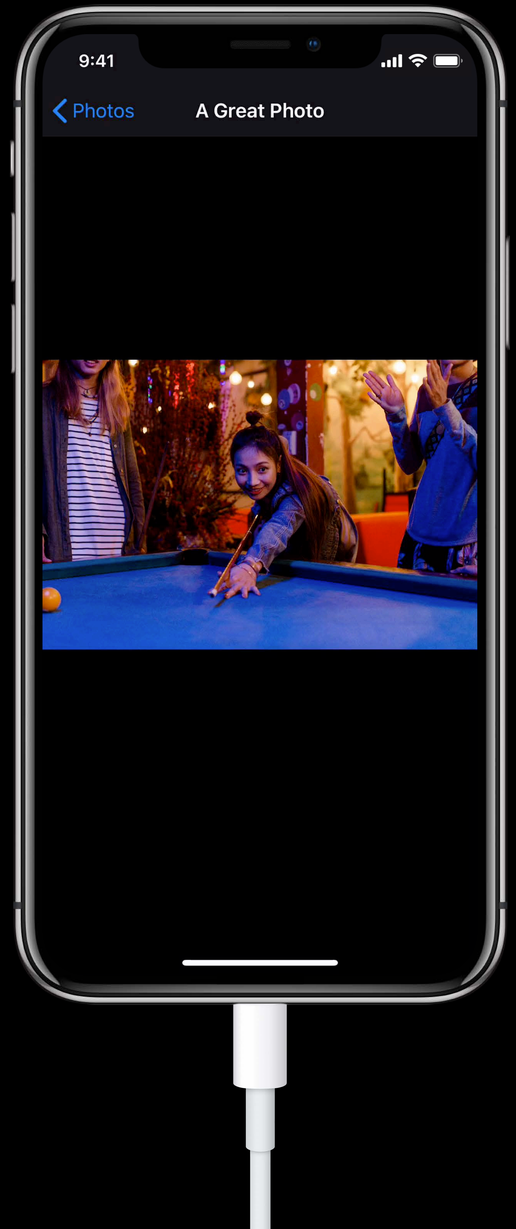
```
// In our Collection View selection callback
if inSingleDisplayMode {
    photoViewController.photo = photo
    navigationController?.pushViewController(photoViewController, animated: true)
} else {
    showOnExternalDisplay(photo)
}
```

Change Your App's Behavior

```
// In our Collection View selection callback
if inSingleDisplayMode {
    photoViewController.photo = photo
    navigationController?.pushViewController(photoViewController, animated: true)
} else {
    showOnExternalDisplay(photo)
}
```

Handle connection changes with graceful transitions.

9:41
Photos
A Great Photo


9:41
Photos
A Great Photo

External Display Support

Add a custom UI for external displays

Consider the different display contexts

Handle connection changes gracefully

Layout-Driven UI

Peter Hajas, iOS System Experience
Ben Fearnley, iOS System Experience

Managing UI complexity.

Recipe for Layout-Driven UI

Recipe for Layout-Driven UI

Find and track state that affects UI

Recipe for Layout-Driven UI

Find and track state that affects UI

Dirty layout when state changes with `setNeedsLayout()`

Recipe for Layout-Driven UI

Find and track state that affects UI

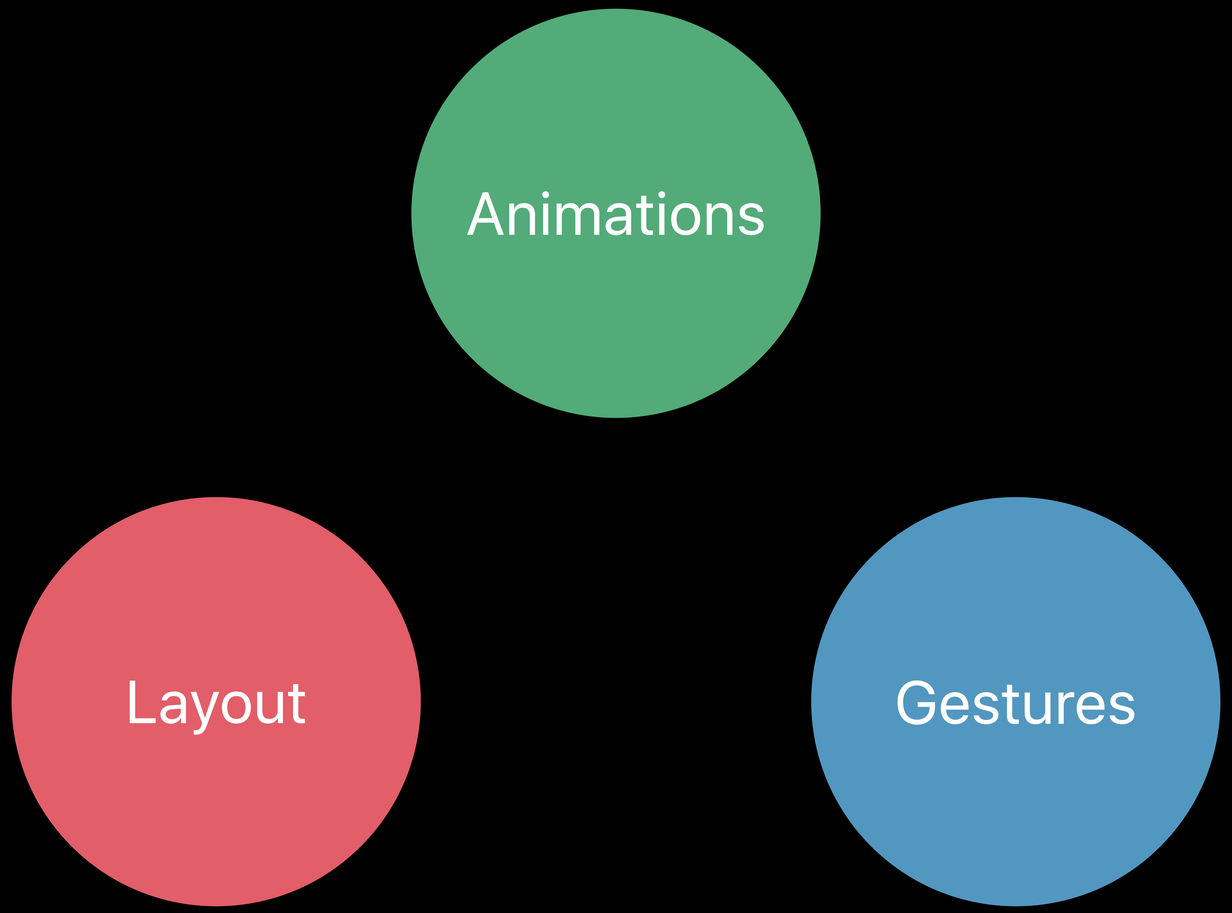
Dirty layout when state changes with `setNeedsLayout()`

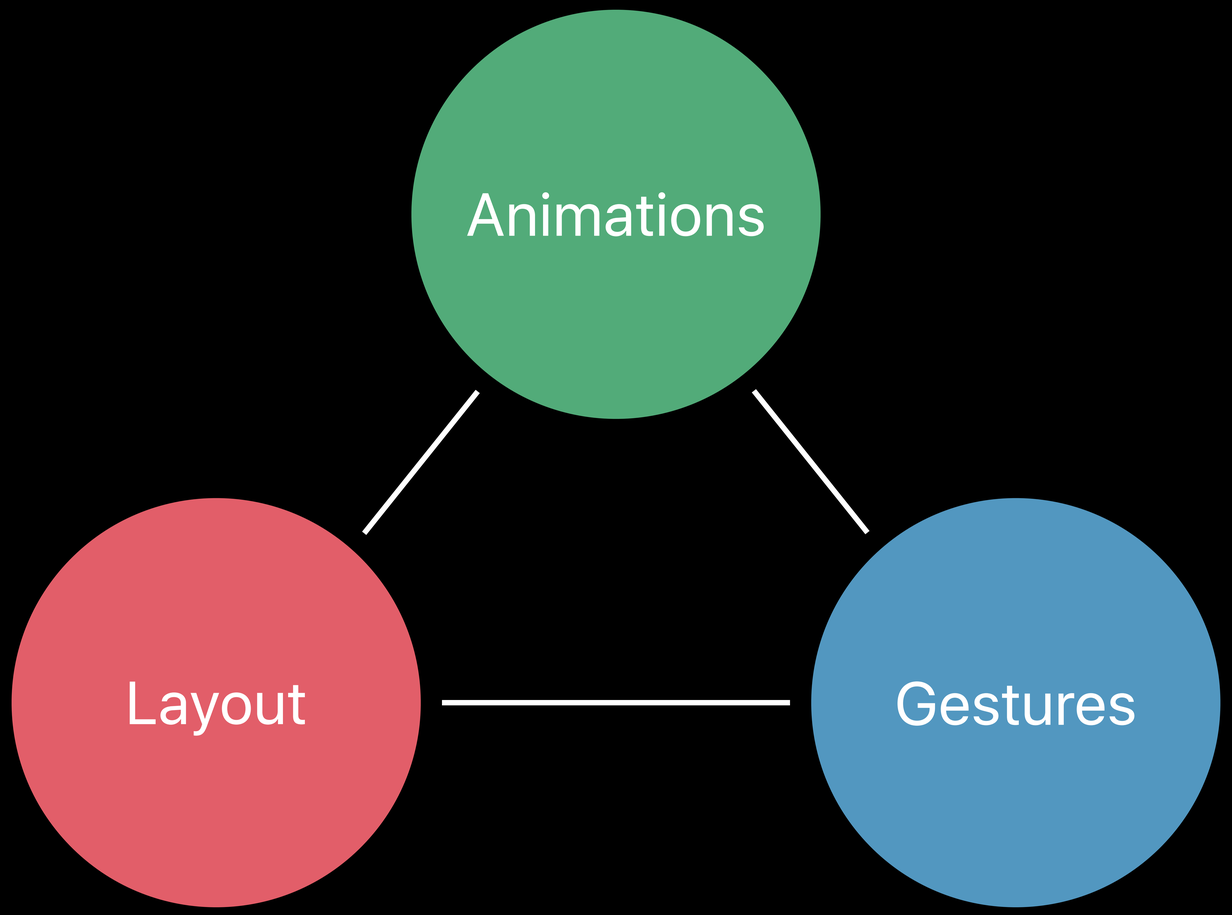
Update UI with state in `layoutSubviews()`

Layout

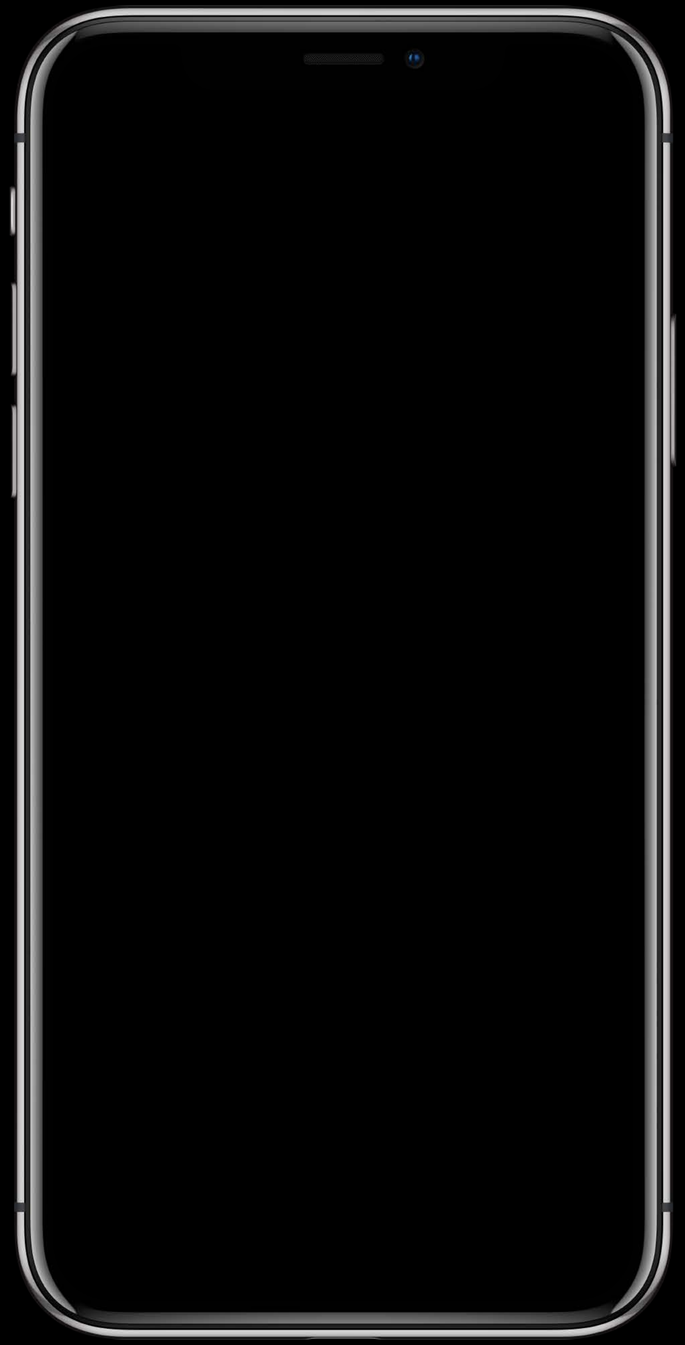
Animations

Gestures


Animations
Layout
Gestures


Animations
Layout
Gestures

Layout

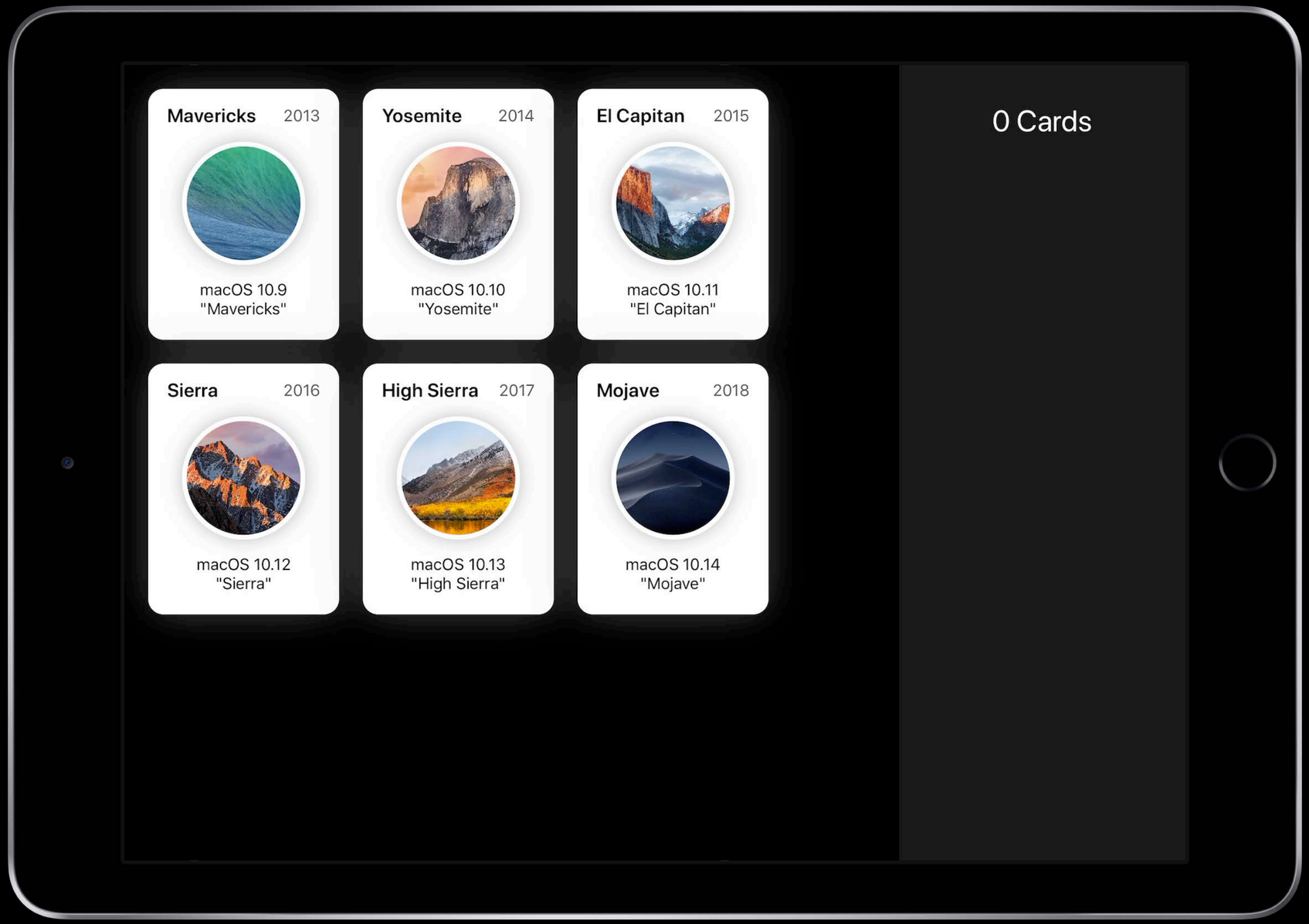


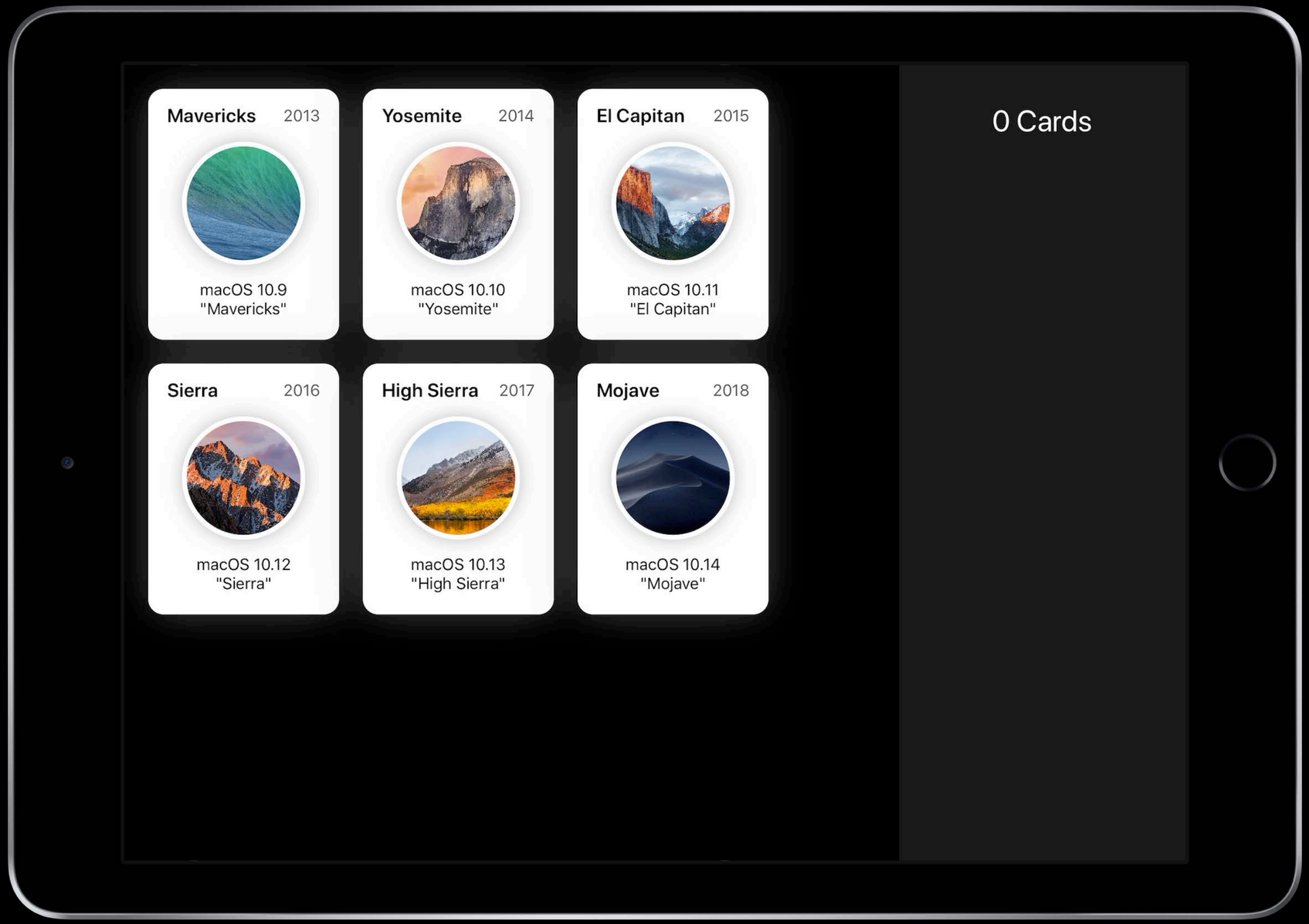
```
class MyView : UIView {
    //...
    let coolView = CoolView()



}
```

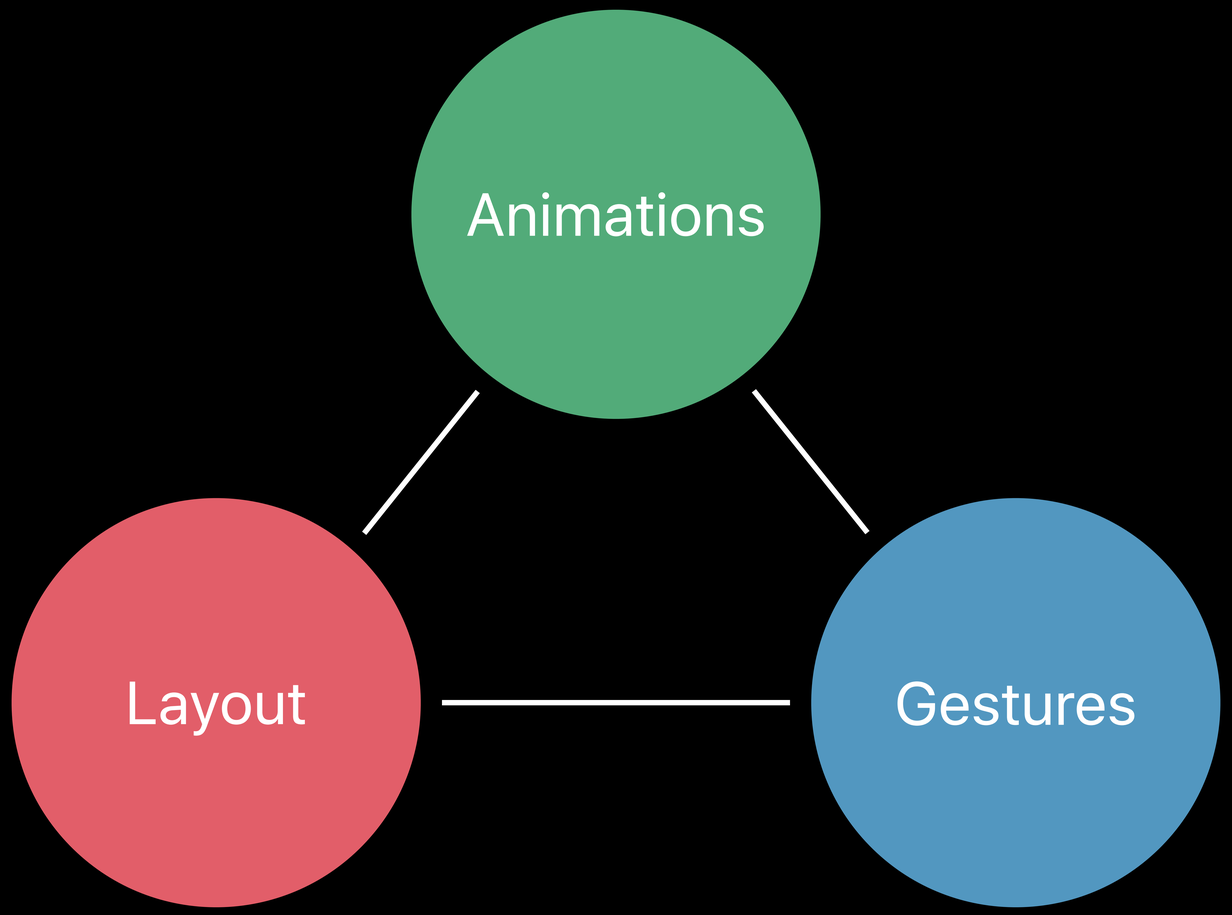
```
class MyView : UIView {
    //...
    let coolView = CoolView()
    var feelingCool = true









}
```

```
class MyView : UIView {
    //...
    let coolView = CoolView()
    var feelingCool = true {
        didSet {
            setNeedsLayout()
        }
    }





}
```

```
class MyView : UIView {
    //...
    let coolView = CoolView()
    var feelingCool = true {
        didSet {
            setNeedsLayout()
        }
    }

    override func layoutSubviews() {
        super.layoutSubviews()
        coolView.isHidden = !feelingCool
    }
}
```


Mavericks 2013
macOS 10.9
"Mavericks"
Yosemite 2014
macOS 10.10
"Yosemite"
El Capitan 2015
macOS 10.11
"El Capitan"
Sierra 2016
macOS 10.12
"Sierra"
High Sierra 2017
macOS 10.13
"High Sierra"
Mojave 2018
macOS 10.14
"Mojave"
0 Cards


Mavericks 2013
macOS 10.9
"Mavericks"
Yosemite 2014
macOS 10.10
"Yosemite"
El Capitan 2015
macOS 10.11
"El Capitan"
Sierra 2016
macOS 10.12
"Sierra"
High Sierra 2017
macOS 10.13
"High Sierra"
Mojave 2018
macOS 10.14
"Mojave"
0 Cards


Animations
Layout
Gestures

Animations

Animation API

UIViewPropertyAnimator

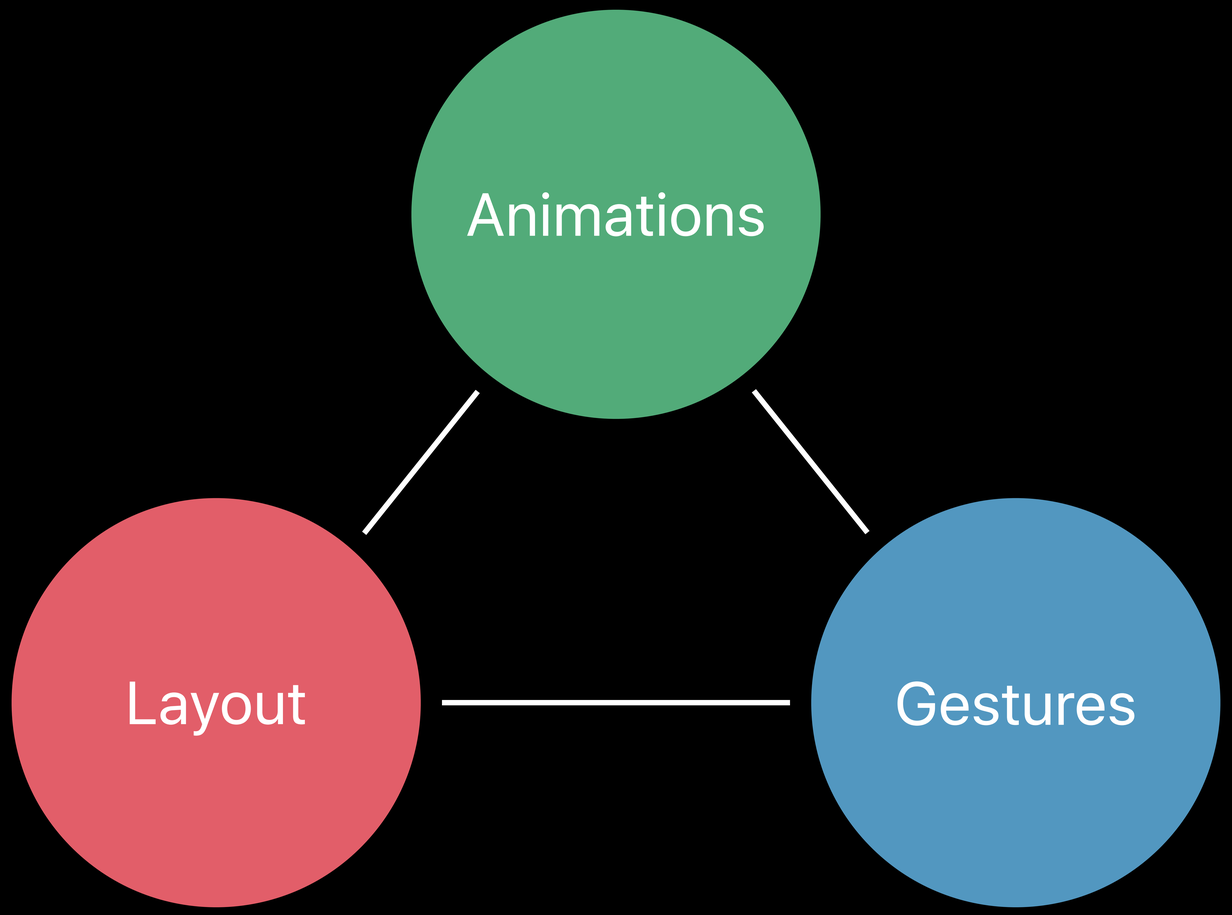
UIView closure API

Animations with Layout-Driven Updates

```
.beginFromCurrentState
```

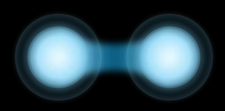
```
var cardsInDeck = [CardView]() {
    didSet {
        setNeedsLayout()
    }
}
```

```
var cardsInDeck = [CardView]() {
    didSet {
        setNeedsLayout()
    }
}

func putCardInDeck(_ card: CardView) {
    cardsInDeck.append(card)
    UIView.animate(withDuration: 0.3,
                   delay: 0,
                   options: [.beginFromCurrentState],
                   animations: {
                    self.layoutIfNeeded()
    }, completion: nil)
}
```

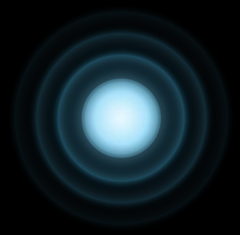

Animations
Layout
Gestures

Gestures

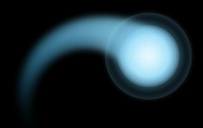
UIGestureRecognizer



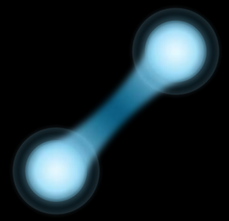
Pan



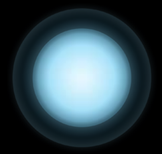
Long Press



Swipe



Pinch

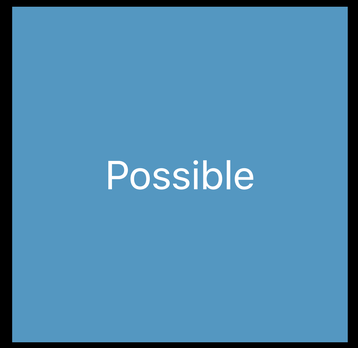


Tap

Rotation

Discrete Gestures

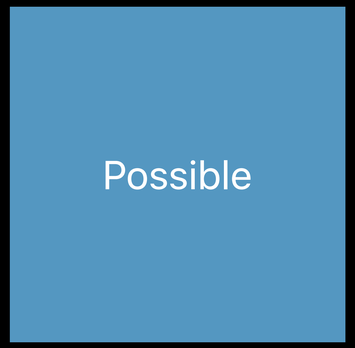
Discrete Gestures



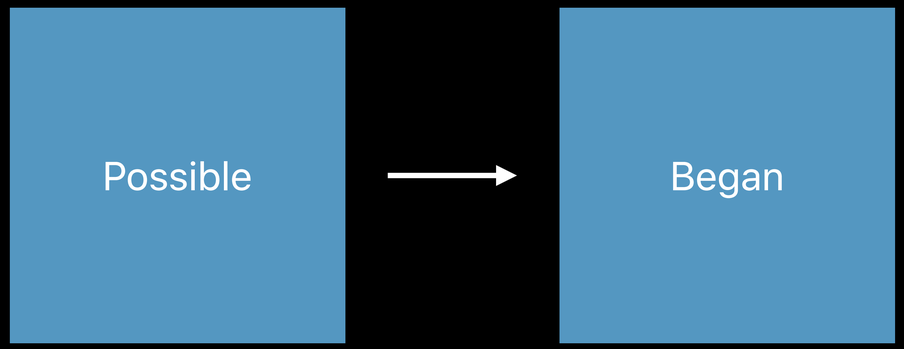
Discrete Gestures

Continuous Gestures

Continuous Gestures

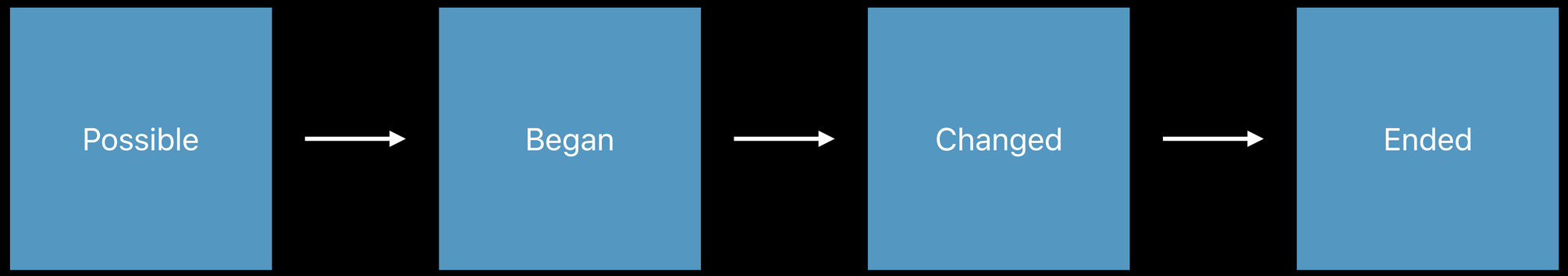


Continuous Gestures



Continuous Gestures

Continuous Gestures



UIPanGestureRecognizer

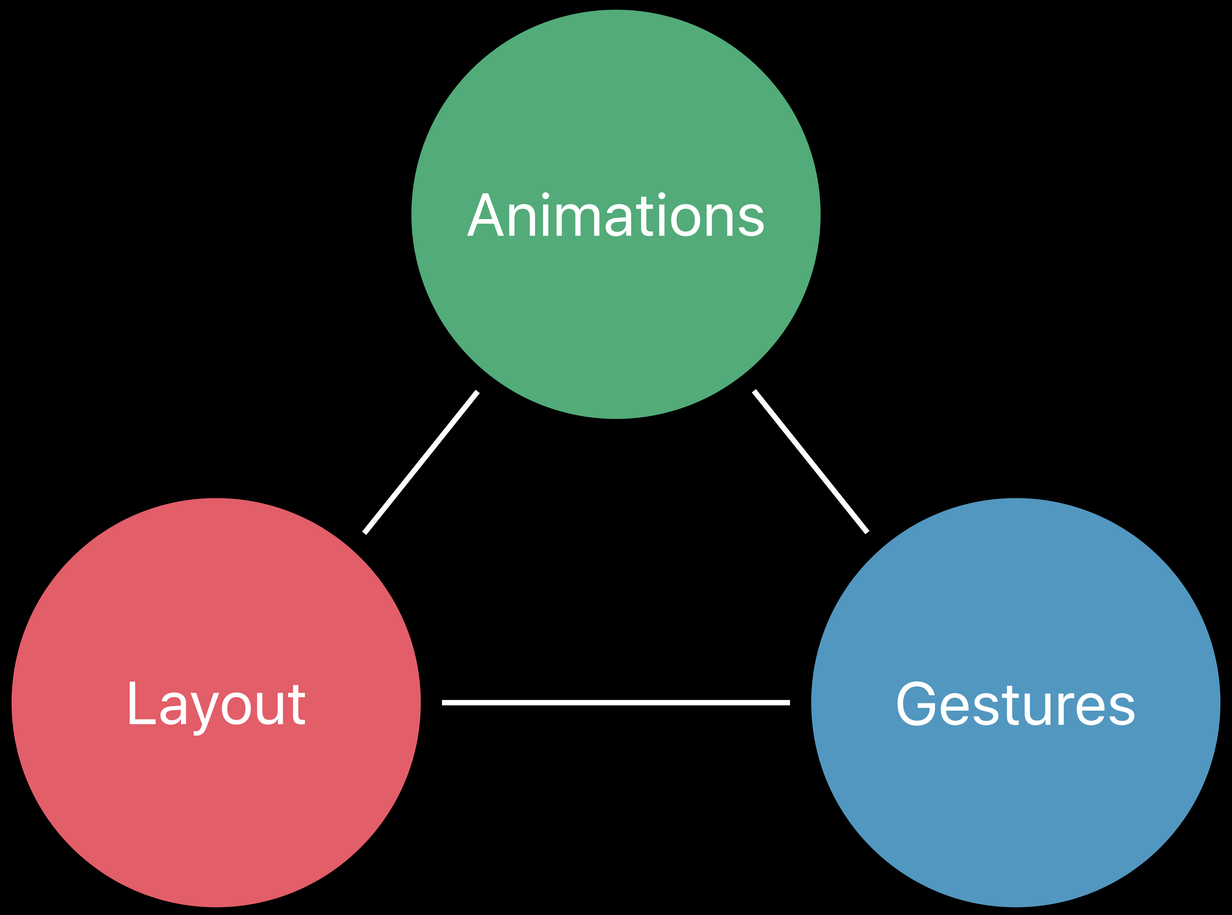
Use `translationInView()` for dragging

Use `velocityInView()` for animation handoffs

```
var cardsToOffsets = [CardView : CGPoint]() {
    didSet {
        setNeedsLayout()
    }
}
```

```
var cardsToOffsets = [CardView : CGPoint]() {
    didSet {
        setNeedsLayout()
    }
}

@objc func handleCardPan(_ pan: UIPanGestureRecognizer) {
    if let card = pan.view as? CardView,
        let currentOffset = cardsToOffsets[card] {
        let translation = pan.translation(in: self)
        cardsToOffsets[card] = CGPoint(x: currentOffset.x + translation.x,
                                       y: currentOffset.y + translation.y)
        pan.setTranslation(.zero, in: self)
    }
}
```

```
var cardsToOffsets = [CardView : CGPoint]() {
    didSet {
        setNeedsLayout()
    }
}

@objc func handleCardPan(_ pan: UIPanGestureRecognizer) {
    if let card = pan.view as? CardView,
        let currentOffset = cardsToOffsets[card] {
        let translation = pan.translation(in: self)
        cardsToOffsets[card] = CGPoint(x: currentOffset.x + translation.x,
                                       y: currentOffset.y + translation.y)
        pan.setTranslation(.zero, in: self)
    }
}

override func layoutSubviews() {
    super.layoutSubviews()
    //...
    for card in cards {
        if let offset = cardsToOffsets[card] {
            card.frame.origin = offset
        }
    }
}
```


Animations
Layout
Gestures

Layout-Driven UI

Find and track state that affects UI

Dirty layout when state changes with `setNeedsLayout()`

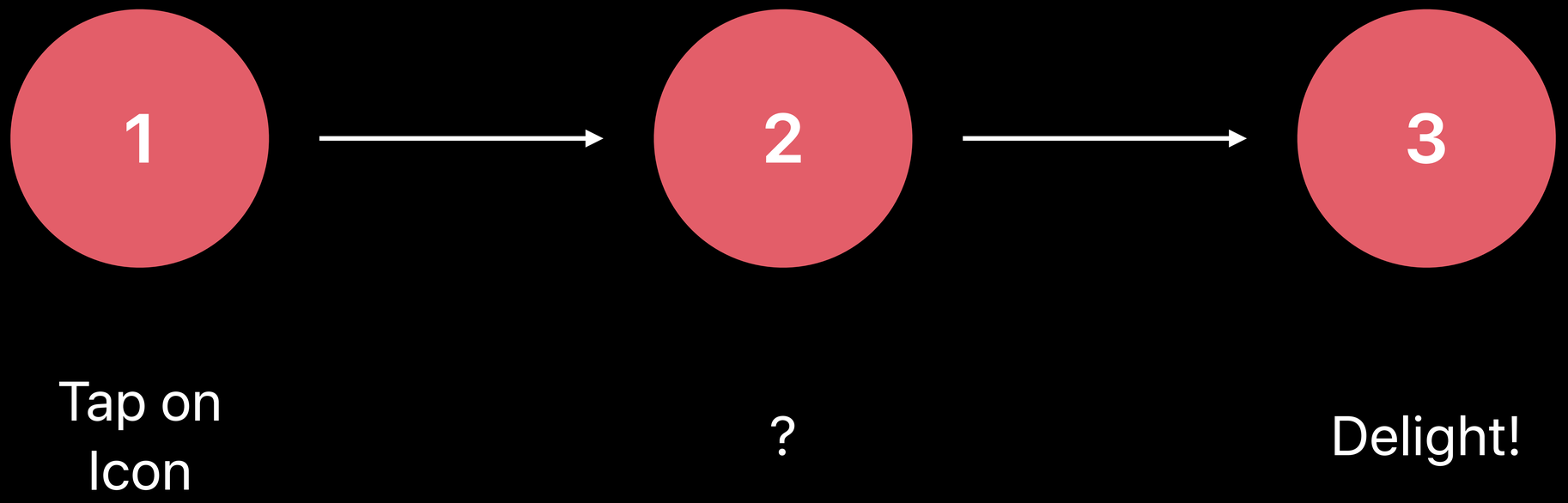
Update UI with state in `layoutSubviews()`

Laser-Fast Launches

Ben Fearnley, iOS System Experience
Peter Hajas, iOS System Experience

The iOS experience is about being responsive.

Get your app responsive quickly.


1
2
3
Tap on Icon
?
Delight!

2

Launch

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

We’ll take care of this.

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

Dynamic Linking

Allocating memory for execution

Linking libraries and frameworks

Initialization of Swift, Objective-C, Foundation, etc.

Static object initialization

40–50 percent of typical app launch time

Dynamic Linking

Avoid code duplication

Limit use of third-party libraries

Avoid static initializers

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

UI Construction

Prepare UI

State restoration

Load preferences

Load model data

UI Construction

Return quickly from

```
application(_:willFinishLaunchingWithOptions:)
application(_:didFinishLaunchingWithOptions:)
applicationDidBecomeActive(_:)
```

Avoid writing to disk

Avoid loading very large data sets

Check database hygiene

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

First Frame

Core Animation renders your first frame

Text drawing

Image loading and decompression

First Frame

Only prepare the UI you need

Avoid hiding views and layers

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

Extended Launch Actions

Tasks deferred to post-launch

App is responsive, but not yet usable

Extended Launch Actions

Prioritize loading content that should be visible at launch

Design for poor network conditions

Anatomy of a Launch

Process Forking

Dynamic Linking

UI Construction

First Frame

Extended Launch Actions

ABM

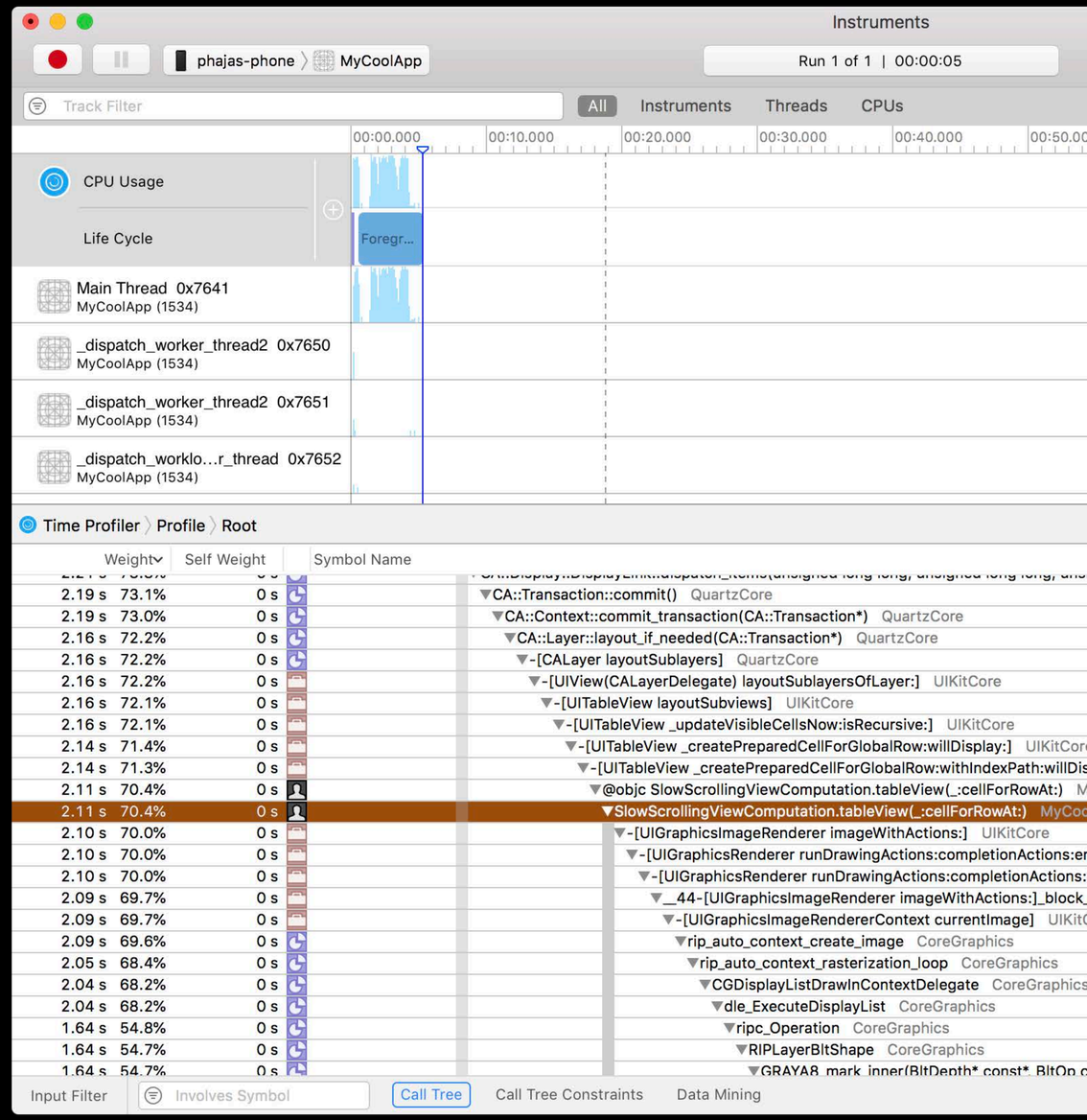
Always Be Measuring

Always Be Measuring

Use Time Profiler to measure your launch

Measure regularly

Use statistical averages



Laser-Fast Launches

Get responsive fast

Use only what you need

Measure, measure, measure

Smooth Scrolling

Peter Hajas, iOS System Experience
Ben Fearnley, iOS System Experience

Why Is My App Slow?

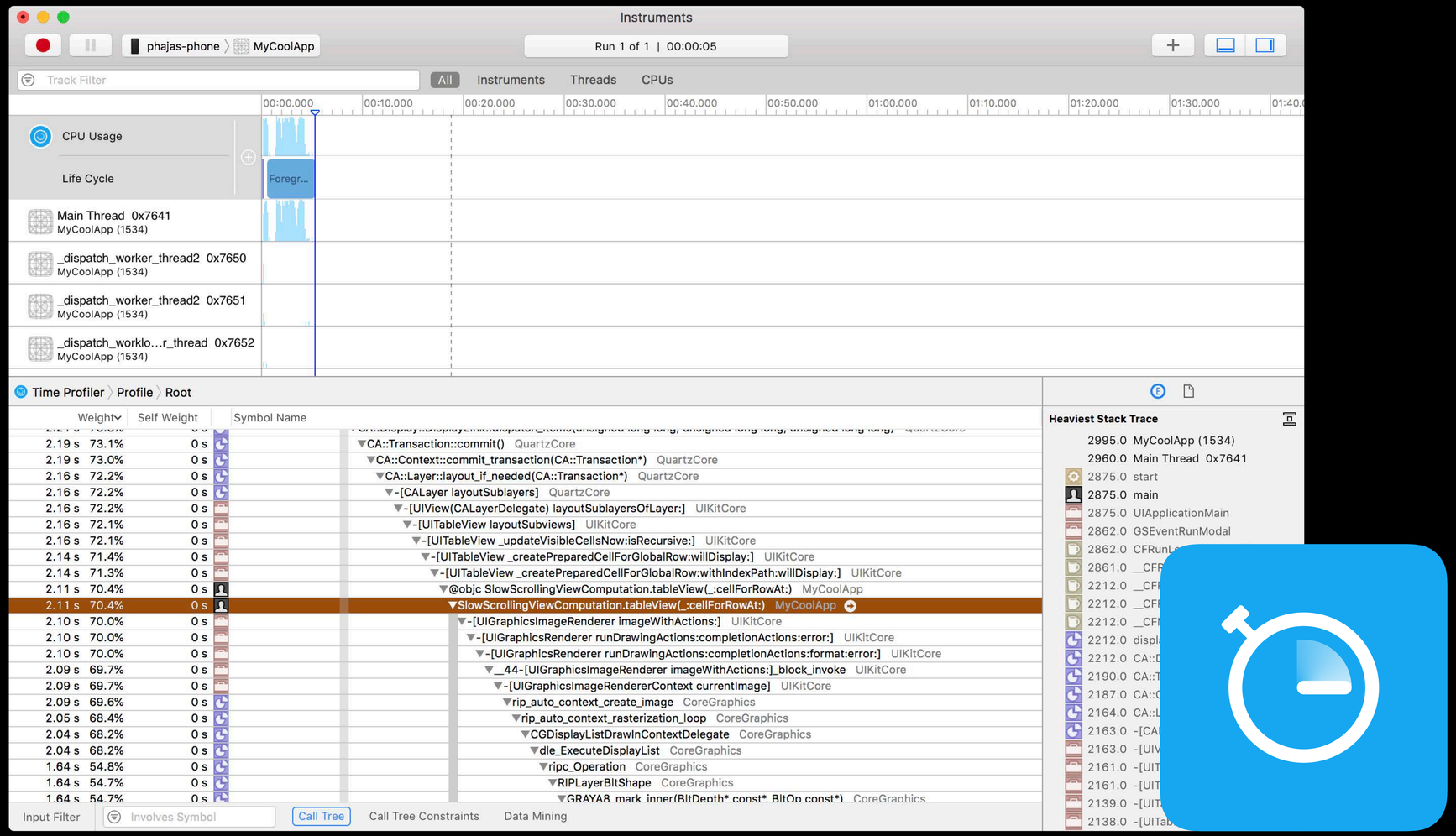
Computation

Graphics

Why Is My App Slow?

Computation

Graphics



Using Time Profiler in Instruments

WWDC 2016

Too Much Computation

`UICollectionView` and `UITableView` prefetching

Push more work to background queues

Work to Push to the Background

Network and file system access

Image drawing

Text sizing

It’s still slow!?

Why Is My App Slow?

Computation

Graphics

Why Is My App Slow?

Computation

Graphics

Why Is My App Slow?

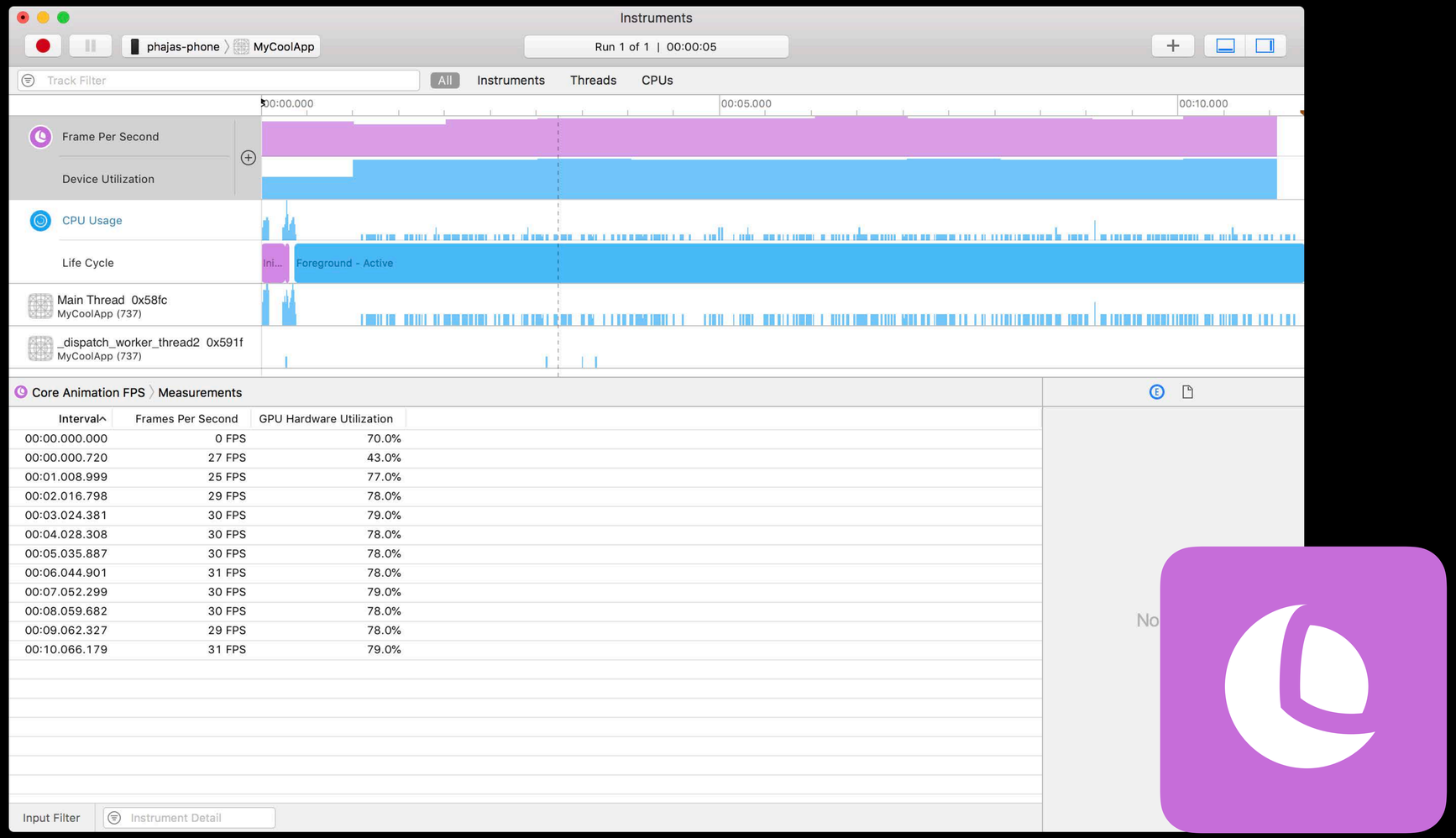
Computation

Graphics

Why Is My App Slow?

Computation

Graphics


Instruments
phajas-phone
MyCoolApp
Run 1 of 1 | 00:00:05
Track Filter
All
Instruments
Threads
CPUs
00:00.000
00:05.000
00:10.000
Frame Per Second
Device Utilization
CPU Usage
Life Cycle
Ini...
Foreground - Active
Main Thread 0x58fc
MyCoolApp (737)
_dispatch_worker_thread2 0x591f
MyCoolApp (737)
Core Animation FPS
Measurements
Interval
Frames Per Second
GPU Hardware Utilization
00:00.000.000 0 FPS 70.0%
00:00.000.720 27 FPS 43.0%
00:01.008.999 25 FPS 77.0%
00:02.016.798 29 FPS 78.0%
00:03.024.381 30 FPS 79.0%
00:04.028.308 30 FPS 78.0%
00:05.035.887 30 FPS 78.0%
00:06.044.901 31 FPS 78.0%
00:07.052.299 30 FPS 79.0%
00:08.059.682 30 FPS 78.0%
00:09.062.327 29 FPS 78.0%
00:10.066.179 31 FPS 79.0%
Input Filter
Instrument Detail

Complex Graphics

Visual effects

Masking

Smooth Scrolling

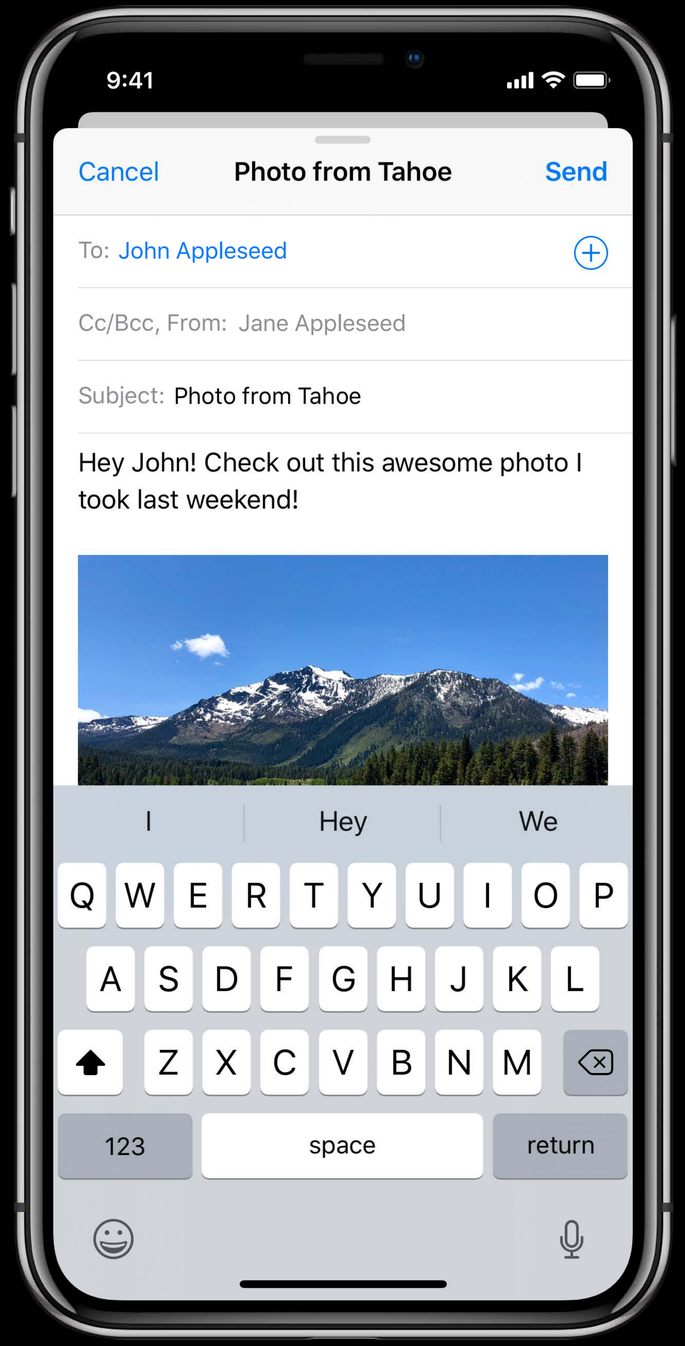
Run Time Profiler and Core Animation instruments on your app

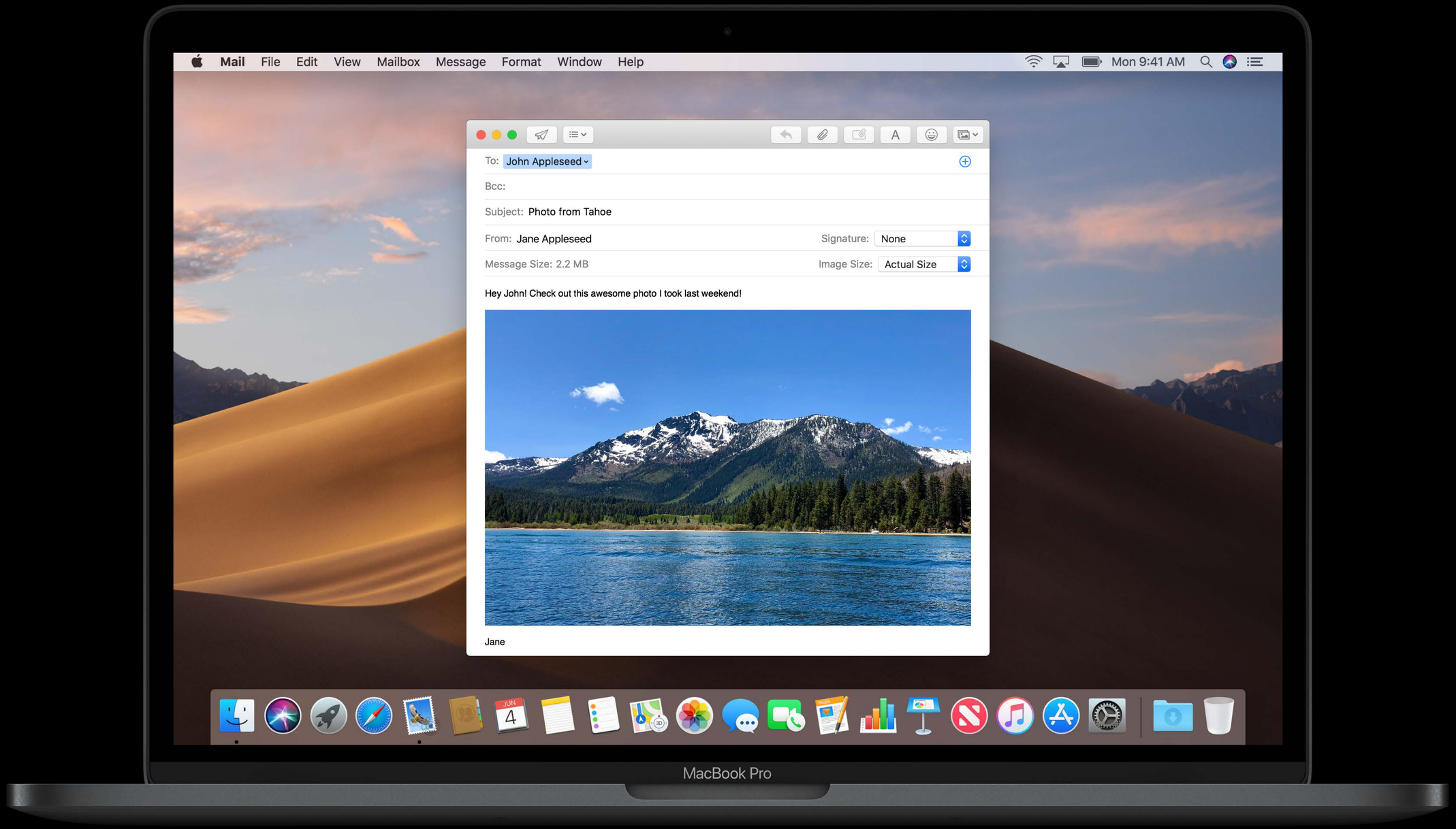
Keep work off the main thread

Use visual effects and masking sparingly

Continuing with Continuity

Ben Fearnley, iOS System Experience
Peter Hajas, iOS System Experience


9:41
Cancel
Photo from Tahoe
Send
To: John Appleseed
Cc/Bcc, From: Jane Appleseed
Subject: Photo from Tahoe
Hey John! Check out this awesome photo I took last weekend!
I
Hey
We
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
return


Mail File Edit View Mailbox Message Format Window Help
Mon 9:41 AM
To: John Appleseed
Bcc:
Subject: Photo from Tahoe
From: Jane Appleseed
Signature: None
Message Size: 2.2 MB
Image Size: Actual Size
Hey John! Check out this awesome photo I took last weekend!
Jane
MacBook Pro

A Magical Experience

Seamlessly transition activities between devices

Works between iOS, macOS, and watchOS

Doesn’t require an Internet connection

It’s easy to set up!

How could you use Handoff?

Messages

Peter
10:09
Hey Peter! Excited about our talk today?
Absolutely! Have you finished the slides yet?

10:09
Peter
Hey Peter! Excited about our talk today?
Absolutely! Have you finished the slides yet?
iMessage
Pay

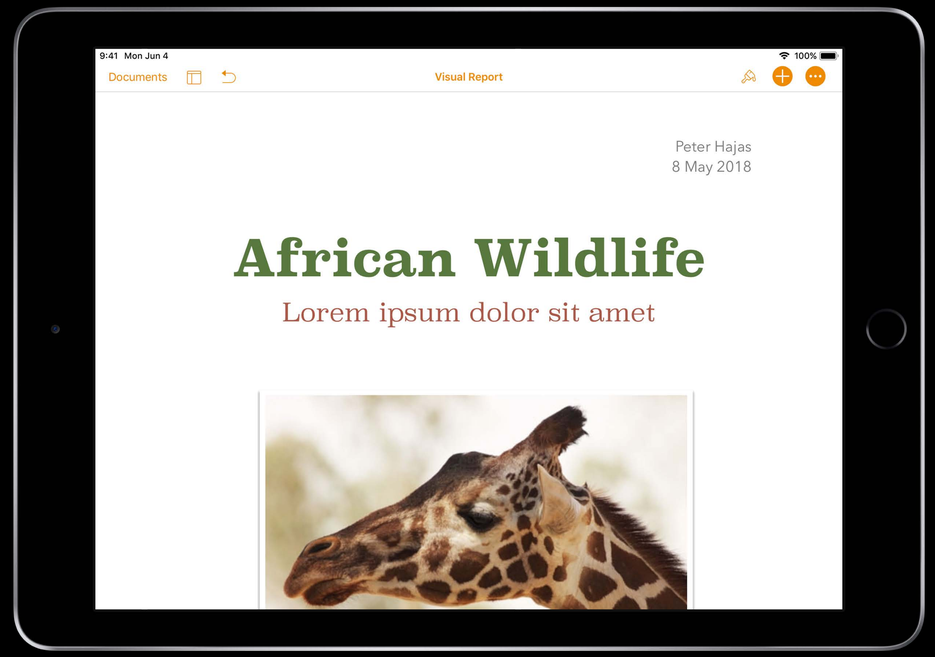
Messages

Peter
10:09
Hey Peter! Excited about our talk today?
Absolutely! Have you finished the slides yet?

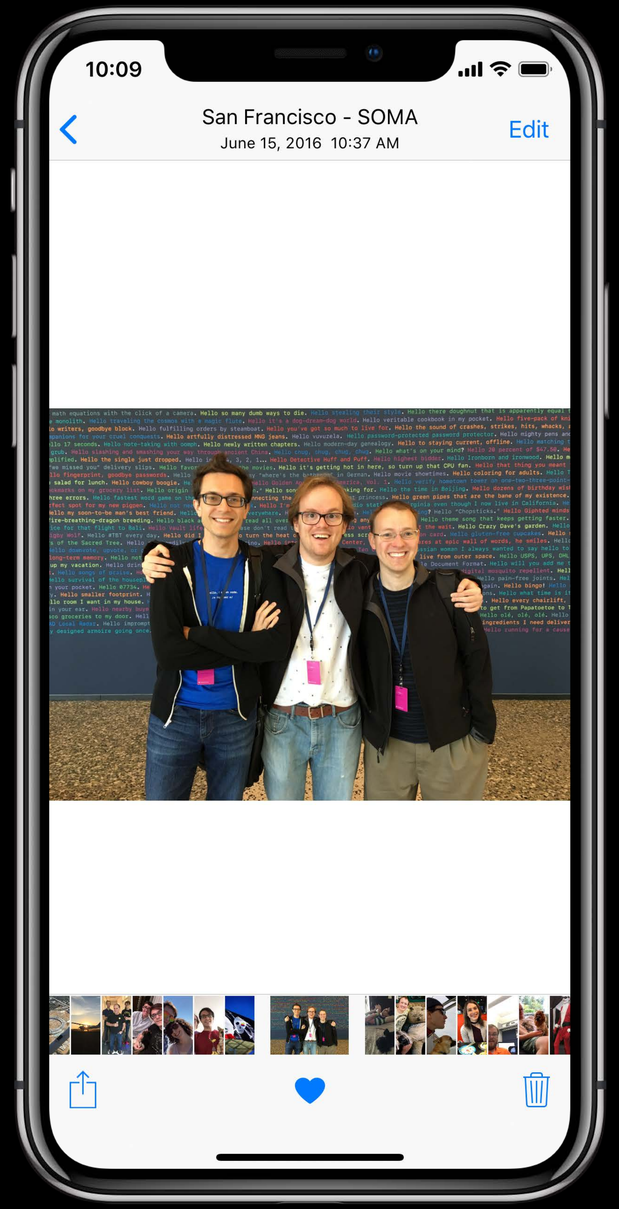
10:09
Peter
Hey Peter! Excited about our talk today?
Absolutely! Have you finished the slides yet?
iMessage
Pay

Documents

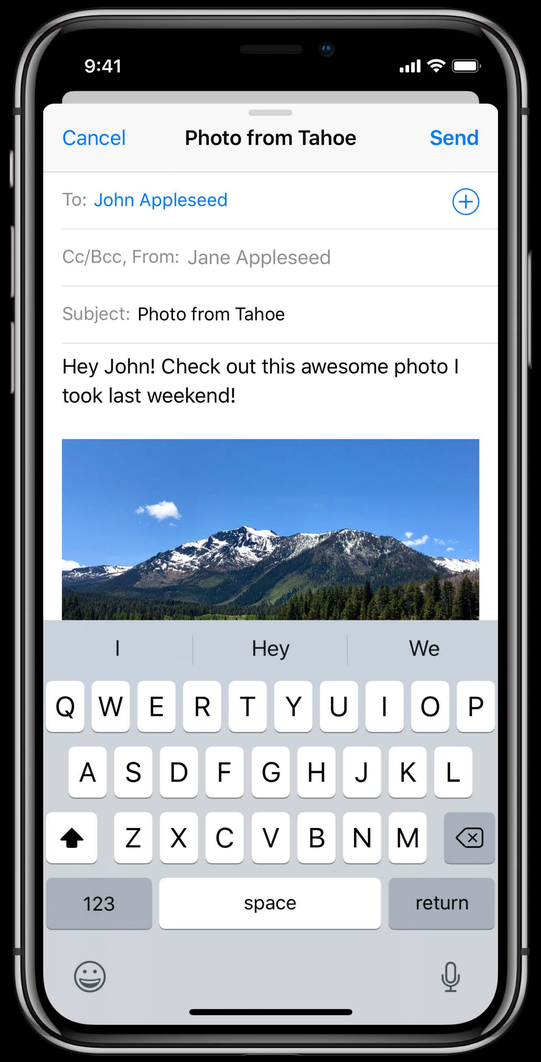
Mon 9:41 AM
Visual Report
125%
Zoom
Add Page
Insert
Table
Chart
Text
Shape
Media
Comment
Collaborate
Format
Document
Peter Hajas
8 May 2018
African Wildlife
Lorem ipsum dolor sit amet
MacBook Pro


9:41 Mon Jun 4
100%
Documents
Visual Report
Peter Hajas
8 May 2018
African Wildlife
Lorem ipsum dolor sit amet

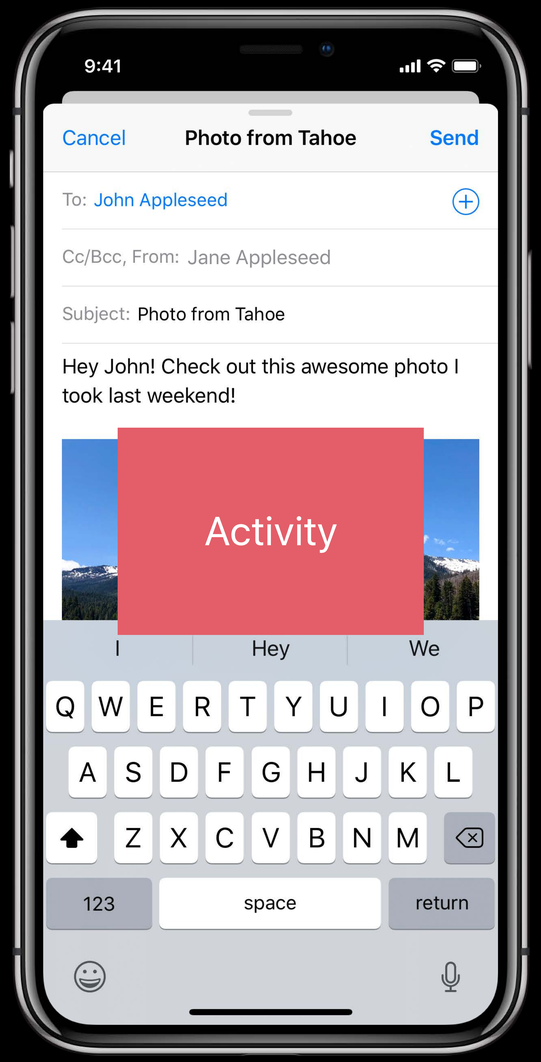
Photos



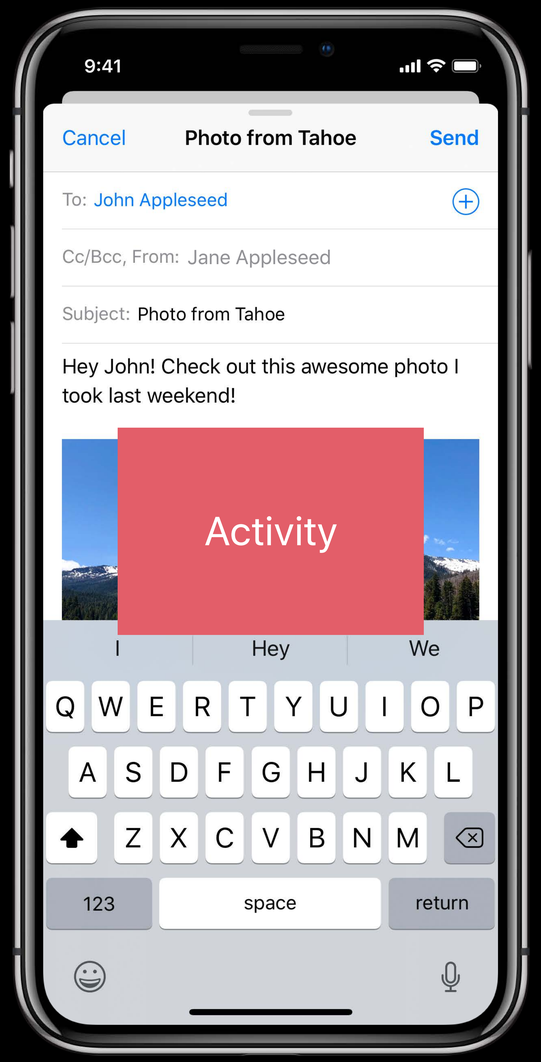
NSUserActivity



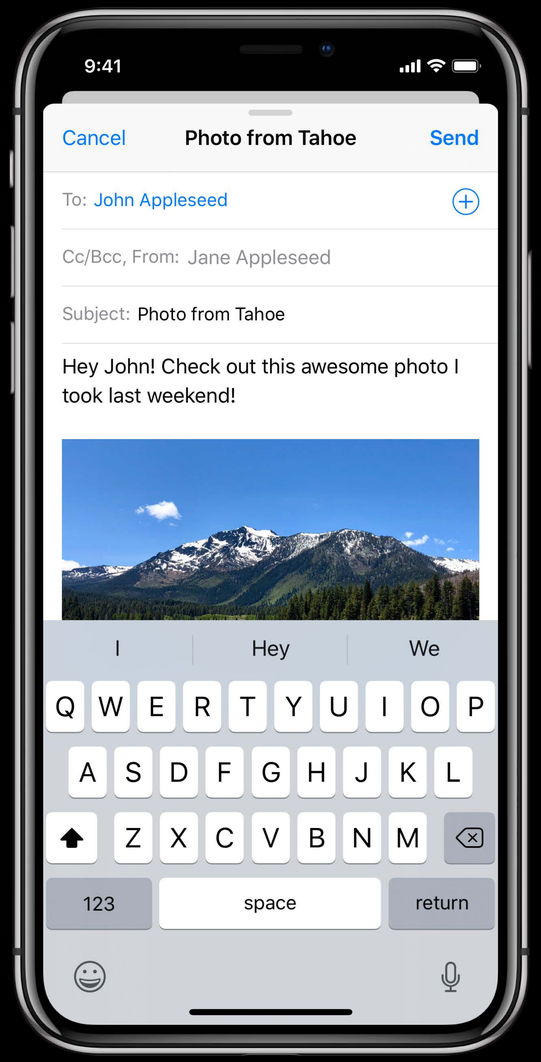
NSUserActivity



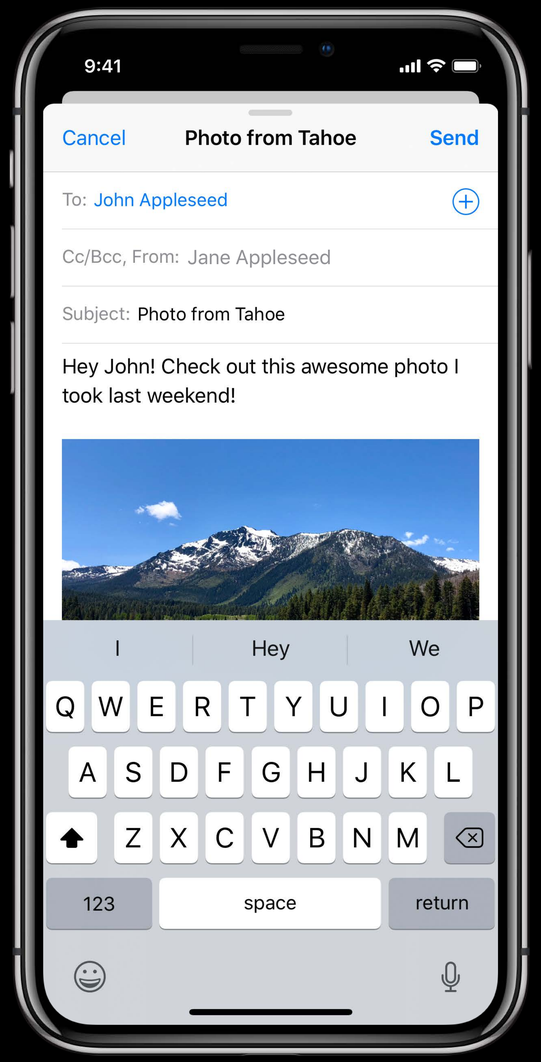
NSUserActivity



NSUserActivity



NSUserActivity



NSUserActivity

Originating Device

```
let activity = NSUserActivity(activityType:"com.apple.developer.video")
activity.title = "Adding Delight to your iOS App"
activity.isEligibleForHandoff = true
activity.userInfo = ["session-id" : "2018-223", "currentTime" : 2340]
userActivity = activity
```

NSUserActivity

Continuing device

App must declare support for activity type

Implement these `UIApplicationDelegate` functions

```
application(_:willContinueUserActivityWithType:)
```

```
application(_:continue:restorationHandler:)
```

Continuation Streams

Set `supportsContinuationStreams` to `true`

Call on the continuing device

```
getContinuationStreams(completionHandler:)
```

Call back on the originating device

```
userActivity(_:didReceive:outputStream:)
```

Document-Based Apps

Get this for free!

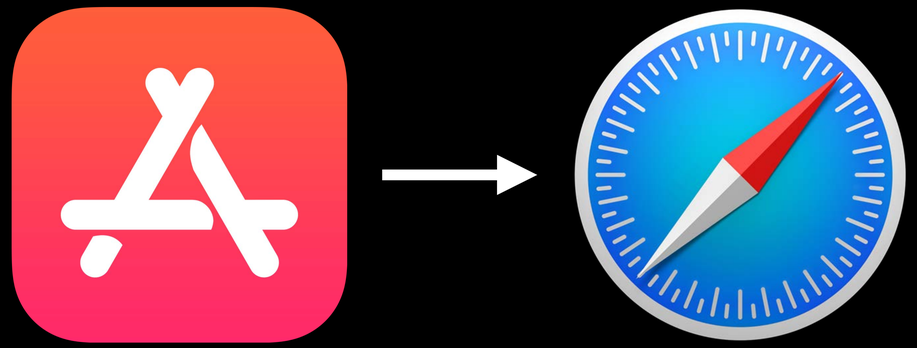
`UIDocument/NSDocument` automatically create `NSUserActivity` objects

Supports documents in iCloud

Configure your `Info.plist`

Native App-to-Web Browser Handoff

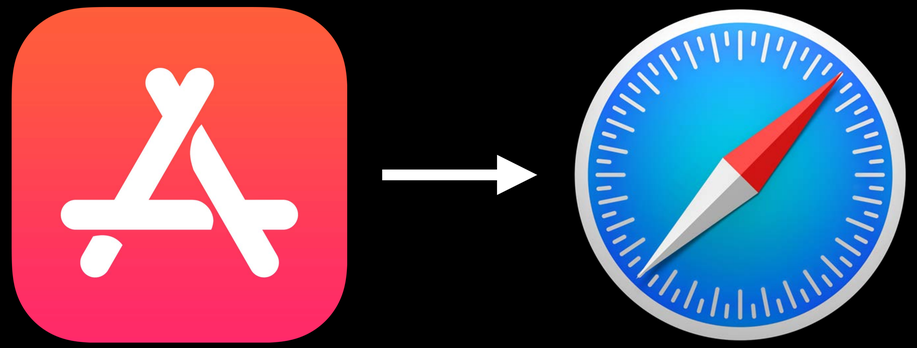
Set `webpageURL` property on the user activity



Web Browser-to-Native App Handoff

Configure list of approved app IDs on web server

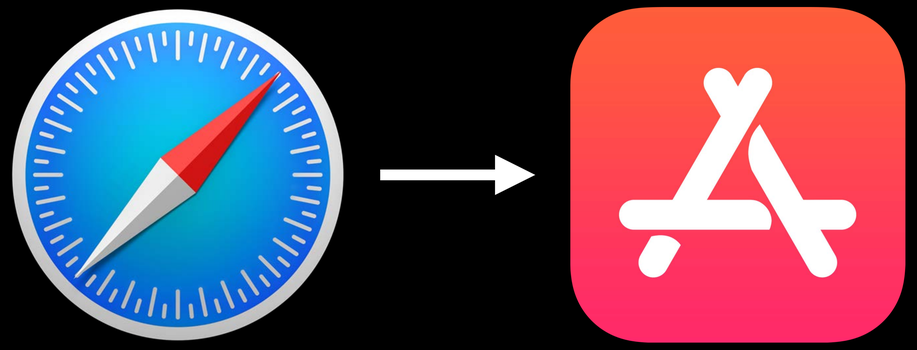
Continuing app must opt in with an associated-domains entitlement



Web Browser-to-Native App Handoff

Configure list of approved app IDs on web server

Continuing app must opt in with an associated-domains entitlement



Continuing with Continuity

Implement Handoff

Delight your users

Integrate with Spotlight search, Siri Shortcuts, and more

Introducing Search APIs	WWDC 2015
Introduction to Siri Shortcuts	WWDC 2018

Debugging like a Pro

Peter Hajas, iOS System Experience
Ben Fearnley, iOS System Experience

Debugging like a Pro

The Detective Mindset

Misplaced Views and View Controllers

State issues

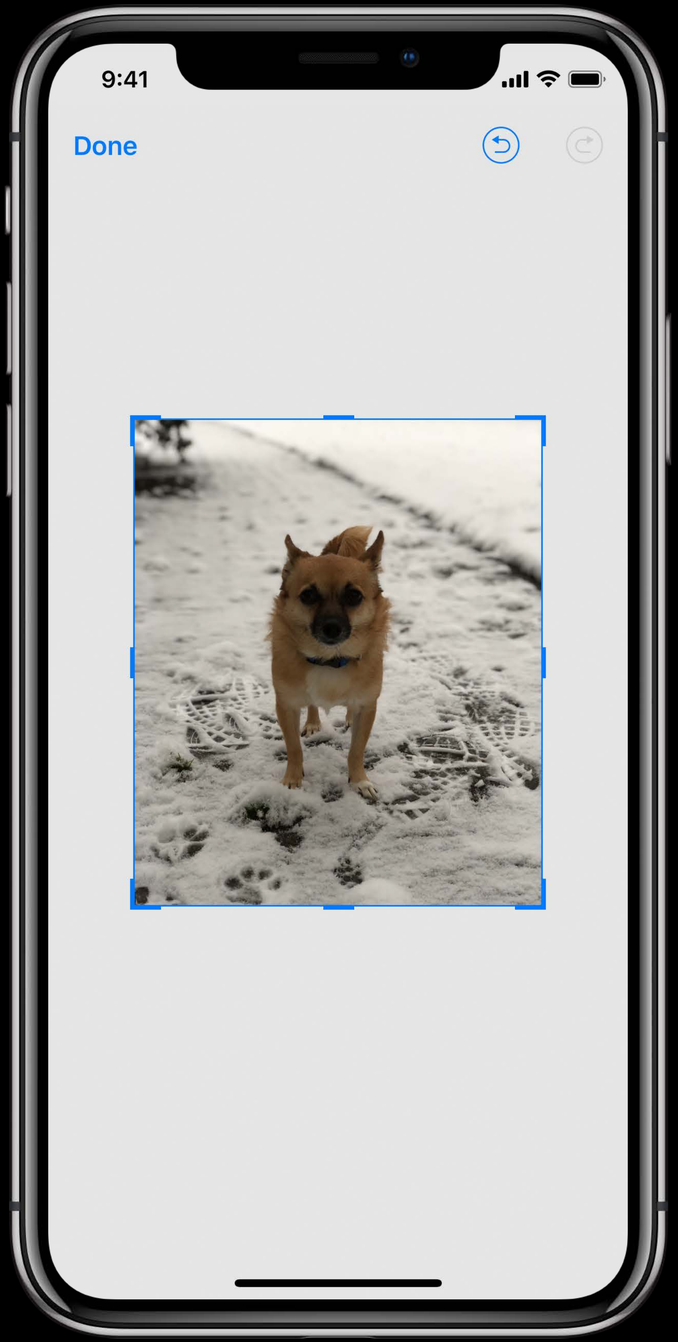
Memory issues

Think like a Detective

Verify your assumptions

Look for clues

Test your hunches


9:41
Done

ScreenshotServicesService 〉 phajas-phone

Running ScreenshotServicesService on phajas-phone

ScreenshotServices 〉 ScreenshotServices 〉 Info.plist 〉 No Selection

ScreenshotServices

Key	Type	Value
Information Property List	Dictionary	(9 items)
Localization native development re...	String	en
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	FMWK
Bundle versions string, short	String	1.0
Bundle version	String	$(CURRENT_PROJECT_VERSION)
Principal class	String	

Filter

ScreenshotServicesService 〉 Thread 1 〉 0 mach_msg_trap

ScreenshotServicesService 〉 phajas-phone

Running ScreenshotServicesService on phajas-phone

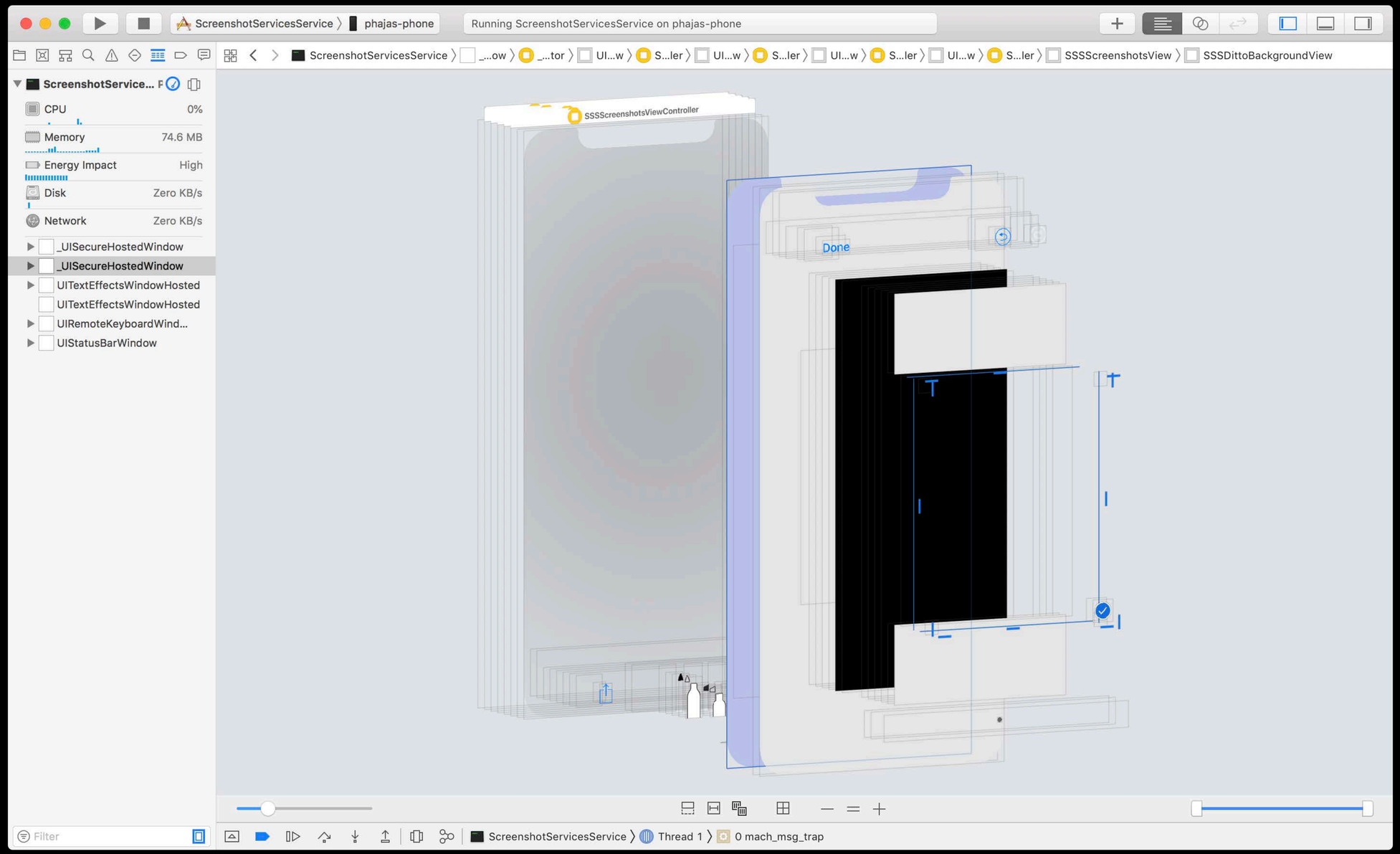
ScreenshotServices 〉 ScreenshotServices 〉 Info.plist 〉 No Selection

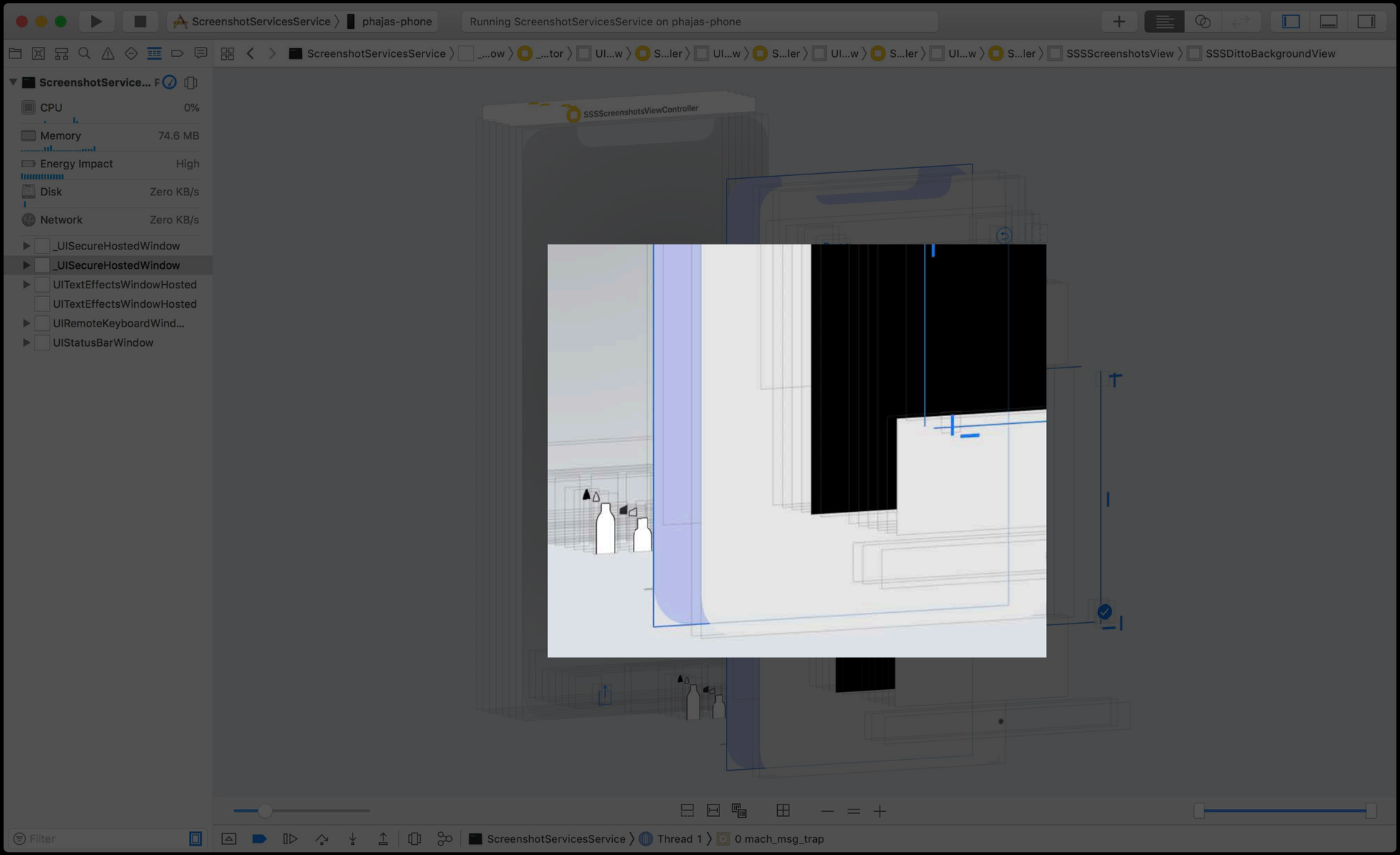
ScreenshotServices

Key	Type	Value
Information Property List	Dictionary	(9 items)
Localization native development re...	String	en
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	FMWK
Bundle versions string, short	String	1.0
Bundle version	String	$(CURRENT_PROJECT_VERSION)
Principal class	String	

Filter

ScreenshotServicesService 〉 Thread 1 〉 0 mach_msg_trap


ScreenshotServicesService
phajas-phone
Running ScreenshotServicesService on phajas-phone
ScreenshotServicesService
_...ow
_...tor
UI...w
S...ler
UI...w
S...ler
UI...w
S...ler
UI...w
S...ler
SSSScreenshotsView
SSSDittoBackgroundView
ScreenshotService...
CPU 0%
Memory 74.6 MB
Energy Impact High
Disk Zero KB/s
Network Zero KB/s
_UISecureHostedWindow
_UISecureHostedWindow
UITextEffectsWindowHosted
UITextEffectsWindowHosted
UIRemoteKeyboardWind...
UIStatusBarWindow
SSSScreenshotsViewController
Done
Filter
ScreenshotServicesService
Thread 1
0 mach_msg_trap


ScreenshotServicesService
phajas-phone
Running ScreenshotServicesService on phajas-phone
ScreenshotServicesService
_...ow
_...tor
UI...w
S...ler
UI...w
S...ler
UI...w
S...ler
UI...w
S...ler
SSSScreenshotsView
SSSDittoBackgroundView
ScreenshotService...
CPU 0%
Memory 74.6 MB
Energy Impact High
Disk Zero KB/s
Network Zero KB/s
_UISecureHostedWindow
_UISecureHostedWindow
UITextEffectsWindowHosted
UITextEffectsWindowHosted
UIRemoteKeyboardWind...
UIStatusBarWindow
SSSScreenshotsViewController
Filter
ScreenshotServicesService
Thread 1
0 mach_msg_trap

Locating a View or View Controller

```
-[UIView recursiveDescription]

-[UIView _parentDescription]

+[UIViewController _printHierarchy]
```

Locating a View or View Controller

```
-[UIView recursiveDescription]

-[UIView _parentDescription]

+[UIViewController _printHierarchy]
```

```
settings set target.language objective-c
```

```
-[UIView recursiveDescription]
```

-[UIView recursiveDescription]

```
(lldb) po [0x129e45380 recursiveDescription]
<SSSScreenshotsView: 0x129e45380; frame = (0 0; 375 812); gestureRecognizers = <NSArray: 0x282c68fc0>; layer = <CALayer: 0x2822f3240>>
   | <SSSShadowView: 0x129e41f60; frame = (0 0; 375 812); alpha = 0; layer = <CALayer: 0x2822f2c80>>
   |    | <_SSSShadowViewInnerViewWithShadow: 0x129e423a0; frame = (0 0; 1875 4060); transform = [5, 0, 0, 5, 0, 0]; layer = <CALayer: 0x2822f2a80>>
   | <SSSDittoBackgroundView: 0x129e427e0; frame = (0 0; 375 812); layer = <CALayer: 0x2822f10c0>>
   | <SSSScrollView: 0x12a81ee00; baseClass = UIScrollView; frame = (0 88; 375 625); gestureRecognizers = <NSArray: 0x282c68d20>; layer = <CALayer: 0x2822f1c00>; contentOffset: {-56, 0}; contentSize: {263, 812}; adjustedContentInset: {0, 56, 0, 56}>
   |    | <UIView: 0x129e41b20; frame = (30 -30; 375 812); hidden = YES; layer = <CALayer: 0x2822f2b20>>
   |    | <SSSScreenshotView: 0x129f51160; frame = (0 28; 263 568); hidden = YES; gestureRecognizers = <NSArray: 0x282cc8ea0>; layer = <CALayer: 0x282563980>>
   |    |    | <SSSScreenshotBorderView: 0x129f1af50; frame = (0 0; 263 568); layer = <CALayer: 0x282563960>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f114b0; frame = (-86 -152; 435 872); alpha = 0; layer = <CALayer: 0x2825635a0>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f16b70; frame = (-81 -147; 425 862); layer = <CALayer: 0x282563720>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f798c0; frame = (-3 109; 269 350.4); alpha = 0; layer = <CALayer: 0x282563900>>
```

-[UIView recursiveDescription]

```
(lldb) po [0x129e45380 recursiveDescription]
<SSSScreenshotsView: 0x129e45380; frame = (0 0; 375 812); gestureRecognizers = <NSArray: 0x282c68fc0>; layer = <CALayer:
0x2822f3240>>
   | <SSSShadowView: 0x129e41f60; frame = (0 0; 375 812); alpha = 0; layer = <CALayer: 0x2822f2c80>>
   |    | <_SSSShadowViewInnerViewWithShadow: 0x129e423a0; frame = (0 0; 1875 4060); transform = [5, 0, 0, 5, 0, 0]; layer =
<CALayer: 0x2822f2a80>>
   | <SSSDittoBackgroundView: 0x129e427e0; frame = (0 0; 375 812); layer = <CALayer: 0x2822f10c0>>
   | <SSSScrollView: 0x12a81ee00; baseClass = UIScrollView; frame = (0 88; 375 625); gestureRecognizers = <NSArray: 0x282c68d20>;
layer = <CALayer: 0x2822f1c00>; contentOffset: {-56, 0}; contentSize: {263, 812}; adjustedContentInset: {0, 56, 0, 56}>
   |    | <UIView: 0x129e41b20; frame = (30 -30; 375 8                                    822f2b20>>
   |    | <SSSScreenshotView: 0x129f51160; frame = (0                                    nizers = <NSArray: 0x282cc8ea0>;
layer = <CALayer: 0x282563980>>
   |    |    | <SSSScreenshotBorderView: 0x129f1af50; frame = (0 0; 263 568); layer = <CALayer: 0x282563960>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f114b0; frame = (-86 -152; 435 872); alpha = 0; layer = <CALayer:
0x2825635a0>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f16b70; frame = (-81 -147; 425 862); layer = <CALayer:
0x282563720>>
   |    |    |    | <_SSSAnimatableBorderCornerRadiusView: 0x129f798c0; frame = (-3 109; 269 350.4); alpha = 0; layer = <CALayer:
0x282563900>>
```

hidden = YES;

-[UIView _parentDescription]

```
(lldb) po [0x129f51160 _parentDescription]
<SSSScreenshotView: 0x129f51160; frame = (0 28; 263 568); hidden = YES; gestureRecognizers = <NSArray:
0x282cc8ea0>; layer = <CALayer: 0x282563980>>
    <SSSScrollView: 0x12a81ee00; baseClass = UIScrollView; frame = (0 88; 375 625); gestureRecognizers =
<NSArray: 0x282c68d20>; layer = <CALayer: 0x2822f1c00>; contentOffset: {-56, 0}; contentSize: {263, 812};
adjustedContentInset: {0, 56, 0, 56}>
        <SSSScreenshotsView: 0x129e45380; frame = (0 0; 375 812); gestureRecognizers = <NSArray: 0x282c68fc0>;
layer = <CALayer: 0x2822f3240>>
            <UIView: 0x129e41060; frame = (0 0; 375 812); autoresize = W+H; layer = <CALayer: 0x2822f1dc0>>
                <UIView: 0x129e0cf90; frame = (0 0; 375 812); autoresize = W+H; tintColor =
UIExtendedSRGBColorSpace 0 0.478431 1 1; layer = <CALayer: 0x2822f06e0>>
                    <UIView: 0x129e4e220; frame = (0 0; 375 812); autoresize = W+H; layer = <CALayer:
0x2822f56e0>>
                        <UIView: 0x129f55570; frame = (0 0; 375 812); autoresize = W+H; layer = <CALayer:
0x282550880>>
                            <_UISecureHostedWindow: 0x129e544a0; frame = (0 0; 375 812); gestureRecognizers =
<NSArray: 0x282ca0a20>; layer = <UIWindowLayer: 0x2822f4f00>>
```

```
+[UIViewController _printHierarchy]
```

+[UIViewController _printHierarchy]

```
(lldb) po [UIViewController _printHierarchy]
<SSSDittoRootViewController 0x129e0def0>, state: disappeared, view: <UIView 0x129e0cf90>
   | <SSSContainerViewController 0x129e4f0c0>, state: appearing, view: <UIView 0x129e41060>
   |    | <SSSScreenshotsViewController 0x129e53d60>, state: appeared, view:
<SSSScreenshotsView 0x129e45380>
```

+[UIViewController _printHierarchy]

```
(lldb) po [UIViewController _printHierarchy]
<SSSDittoRootViewController 0x129e0de                                    iew 0x129e0cf90>
   | <SSSContainerViewController 0x12                                    UIView 0x129e41060>
   |    | <SSSScreenshotsViewControll                                    iew:
<SSSScreenshotsView 0x129e45380>
```

state: appearing

Debugging State Issues

LLDB's `expr` command

Run arbitrary code in the debugger!

```
expr myStruct.doSomething()
```

Debugging with LLDB	WWDC 2012
Advanced Swift Debugging in LLDB	WWDC 2014

dump

dump

```
(lldb) expr dump(card)
▽ <Fluid.CardView: 0x10091dd70; frame = (475 400; 200 350); gestureRecognizers = <NSArray: 0x280c668b0>; layer = <CALayer: 0x28021d0e0>> #0
  - super: UIView
    - super: UIResponder
      - super: NSObject
  - backgroundView: <UIView: 0x10091df80; frame = (0 0; 200 350); layer = <CALayer: 0x28021d220>> #1
    - super: UIResponder
      - super: NSObject
  - nameLabel: <UILabel: 0x10091e160; frame = (20 16; 160 24); text = 'John'; userInteractionEnabled = NO; layer = <_UILabelLayer: 0x28211f160>> #2
    - super: UIView
      - super: UIResponder
        - super: NSObject
  - imageView: <UIImageView: 0x10091e450; frame = (20 16; 160 160); clipsToBounds = YES; opaque = NO; userInteractionEnabled = NO; layer = <CALayer:
0x28021d160>> - (null) #3
    - super: UIView
      - super: UIResponder
        - super: NSObject
  - descriptionLabel: <UILabel: 0x10091e680; frame = (20 192; 160 81.5); text = 'Mojave'; userInteractionEnabled = NO; layer = <_UILabelLayer:
0x28211f250>> #4
    - super: UIView
      - super: UIResponder
        - super: NSObject
```

dump

```
(lldb) expr dump(card)
▽ <Fluid.CardView: 0x10091dd70; frame = (475 400; 200 350); gestureRecognizers = <NSArray: 0x280c668b0>; layer = <CALayer: 0x28021d0e0>> #0
  - super: UIView
    - super: UIResponder
      - super: NSObject
  - backgroundView: <UIView: 0x10091df80; frame = (0 0; 200 350); layer = <CALayer: 0x28021d220>> #1
    - super: UIRespon
      - super: NSObje
  - nameLabel: <UILab
    - super: UIView
      - super: UIResponder
        -
  - imageV
0x28021d16
    - super: UIView
      - super: UIResponder
        - super: NSObject
  - descriptionLabel: <UILabel: 0x10091e680; frame = (20 192; 160 81.5); text = 'Mojave'; userInteractionEnabled = NO; layer = <_UILabelLayer:
0x28211f250>> #4
    - super: UIView
      - super: UIResponder
        - super: NSObject
```

```
<UILabel: 0x10091e160; frame = (20 16; 160 24);
```

```
<UIImageView: 0x10091e450; frame = (20 16; 160 160);
```

dump

```
(lldb) expr dump(card)
▽ <Fluid.CardView: 0x10091dd70; frame = (475 400; 200 350); gestureRecognizers = <NSArray: 0x280c668b0>; layer = <CALayer: 0x28021d0e0>> #0
  - super: UIView
    - super: UIResponder
      - super: NSObject
  - backgroundView: <UIView: 0x10091df80; frame = (0 0; 200 350); layer = <CALayer: 0x28021d220>> #1
    - super: UIRespon
      - super: NSObje
  - nameLabel: <UILab
    - super: UIView
      - super: UIResponder
        -
  - imageV
0x28021d16
    - super: UIView
      - super: UIResponder
        - super: NSObject
  - descriptionLabel: <UILabel: 0x10091e680; frame = (20 192; 160 81.5); text = 'Mojave'; userInteractionEnabled = NO; layer = <_UILabelLayer:
0x28211f250>> #4
    - super: UIView
      - super: UIResponder
        - super: NSObject
```

```
<UILabel: 0x10091e160; frame = (20 16; 160 24);
```

```
<UIImageView: 0x10091e450; frame = (20 16; 160 160);
```

-[NSObject _ivarDescription]

-[NSObject _ivarDescription]

```
(lldb) po [0x129f74c70 _ivarDescription]
<SSSScreenshotView: 0x129f74c70>:
in SSSScreenshotView:
    _cropController (SSSCropController*): <SSSCropController: 0x280743a20>
    _cropEnabled (BOOL): NO
    _borderView (SSSScreenshotBorderView*): <SSSScreenshotBorderView: 0x129f74020>
    _checkView (SSSCheckView*): <SSSCheckView: 0x129f8a1e0>
    _tapGesture (UITapGestureRecognizer*): <UITapGestureRecognizer: 0x281f24000>
```

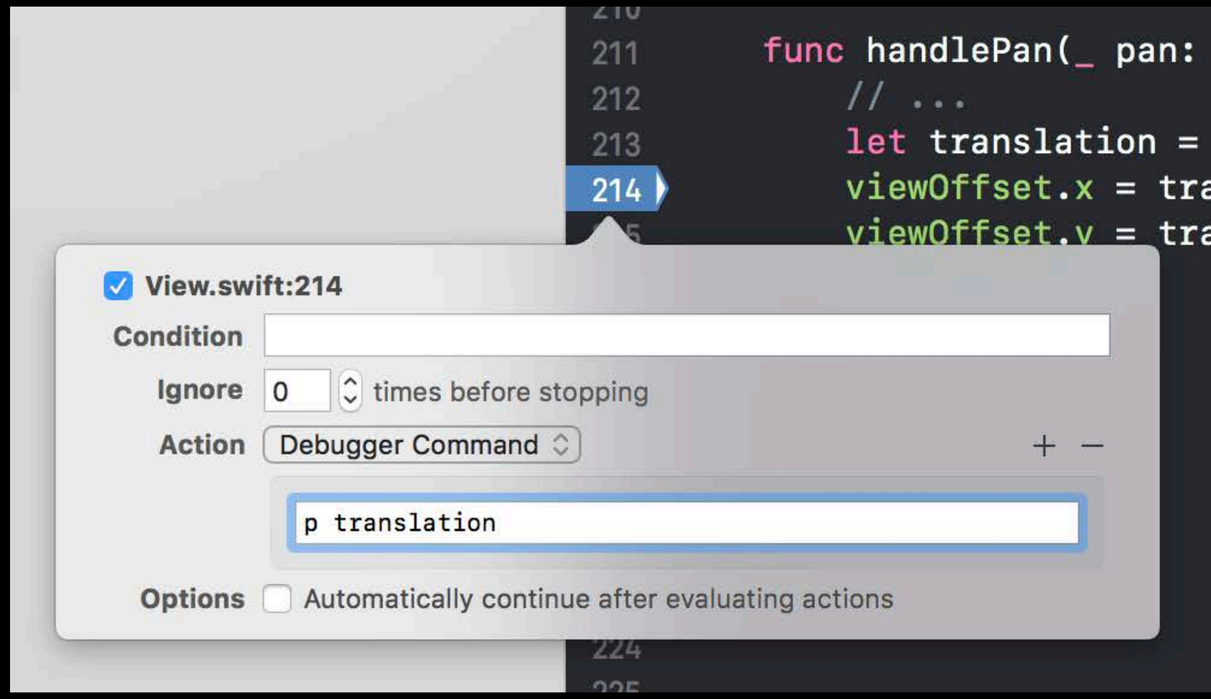
-[NSObject _ivarDescription]

```
(lldb) po [0x129f74c70 _ivarDescription]
<SSSScreenshotView: 0x129f74c70>:
in SSSScreenshotView:
                                                    Controller: 0x280743a20>
  _cropEnabled (BOOL): NO
                                                    reenshotBorderView: 0x129f74020>
    _checkView (SSSCheckView*): <SSSCheckView: 0x129f8a1e0>
    _tapGesture (UITapGestureRecognizer*): <UITapGestureRecognizer: 0x281f24000>
```

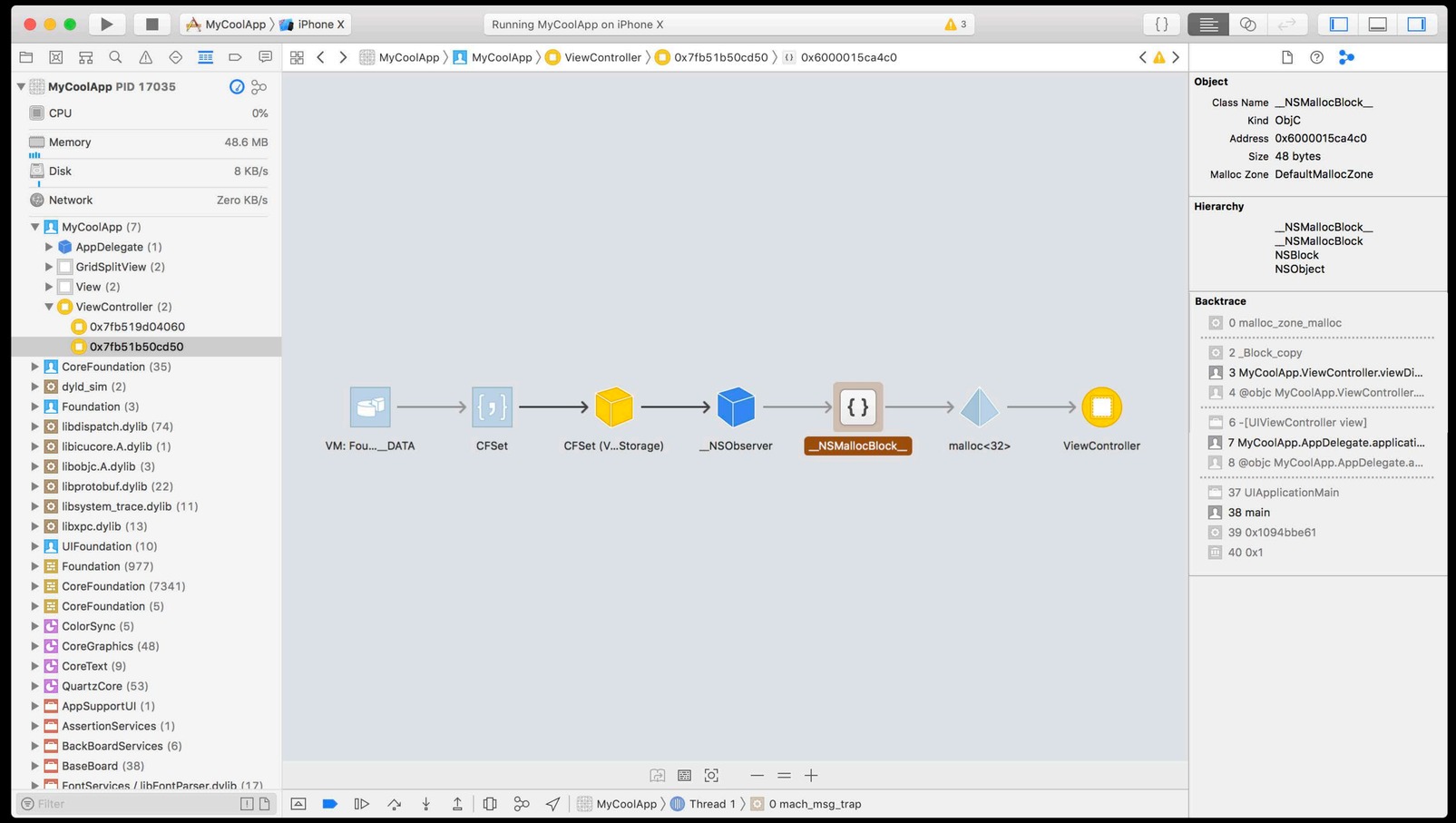
Breakpoints

Pause the program

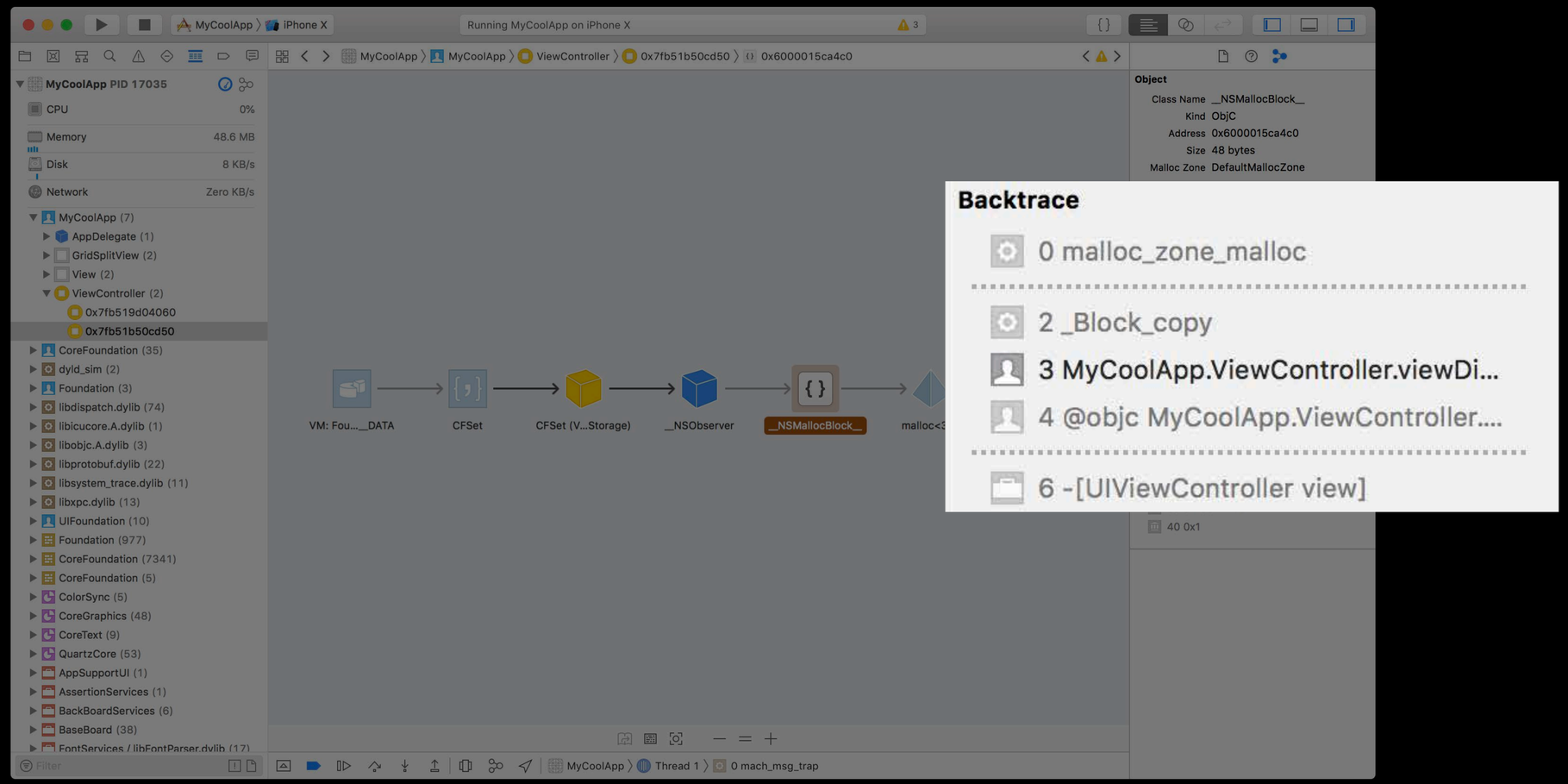
Commands



Debugging Memory Issues



Debugging Memory Issues



Debugging like a Pro

Think like a detective!

Xcode View Debugger and Memory Debugger

LLDB `expr` and `dump`

External Display Support

Layout-Driven UI

Laser-Fast Launches

Smooth Scrolling

Continuing with Continuity

Debugging Like a Pro

More Information

http://developer.apple.com/wwdc2018/233

WWDC18